Valid inequalities for the k-Color Shortest Path Problem

Rafael Castro de Andrade*

Universidade Federal do Ceará, Departamento de Estatística e Matemática Aplicada, Campus do Pici, bloco 910, Fortaleza, Ceará, Brasil. Postal Code 60.440-900.

Emanuel Elias Silva Castelo

Universidade Federal do Ceará, Mestrado e Doutorado em Ciência da Computação, Campus do Pici, bloco 910, Fortaleza, Ceará, Brasil. Postal Code 60.440-900.

Rommel Dias Saraiva

Universidade de Fortaleza, Laboratório de Ciência de Dados e Inteligência Artificial, Av. Washington Soares 1321, Sl. M04, Postal Code 60811-905.

Abstract

Given a digraph $D = (V, A)$ where each arc $(i, j) \in A$ has a cost $d_{ij} \in \mathbb{R}_+$ and a color $c(i, j)$, a positive integer k, and vertices $s, t \in V$, the k-Color Shortest Path Problem consists in finding a path from s to t of minimum cost while using at most k distinct arc colors. We propose valid inequalities for the problem that proved to strengthen the linear relaxation of an existing Integer Linear Programming formulation for the problem. One exponential set of valid inequalities defines a new formulation for the problem that is solved by using a branch-and-cut algorithm. We introduce more challenging instances for the problem and present numerical experiments for both the benchmark and the new instances. Finally, we evaluate the individual and the collective use of the valid inequalities. Computational results for the proposed ideas and for existing solution approaches for the problem showed the effectiveness of the new inequalities in handling the new instances, both in terms of execution times and improvement of the linear relaxed solutions.

Keywords: combinatorial optimization, k-color shortest path problem, valid inequalities.

*Corresponding author (rafael.andrade@ufc.br)

Email addresses: `rafael.andrade@ufc.br` (Rafael Castro de Andrade), `emanuelcastelo@lia.ufc.br` (Emanuel Elias Silva Castelo), `rommelds@unifor.br` (Rommel Dias Saraiva)

1. Introduction

Some optimization problems defined on edge-colored graphs encode qualitative information using colors (or labels), which can represent different alternatives of transportation in multi-modal networks or types of connections in computer systems. Examples of applications can be found in genetic and molecular biology (Dorninger, 1994; Pevzner, 1995), design of reliable networks (Yuan et al., 2005; Chang and Shing-Jiuan, 1997) and light paths in Wavelength-Division Multiplexing (WDM) optical networks (Santos et al., 2016). We also refer to paths with the minimum number of obstacles in robotics (Eiben and Kanj, 2020), which aims at finding a path that does not cross more than a given number of different obstacles. In telecommunications, there are problems related to shared risk link groups (Shen et al., 2005; Craveirinha et al., 2023), which consider sets of links that are likely to fail concurrently as they share physical resources.

In the literature, we find related works in this context. The Minimum Color Path Problem (MCPP), for instance, consists in finding a path between vertices s and t (an (s,t)-path, for short) with the minimum number of distinct colors. Although originally proposed for reliable networks (Yuan et al., 2005), the problem also has applications in robotics, where one wants to find a path between two points while traversing the minimum number of obstacles. To handle the MCPP, the authors developed an $O(n^{2/3})$-approximation algorithm, two greedy heuristics and an Integer Linear Programming (ILP) formulation.

Another example is the Single k-Multicolor Path Problem (SMPP), which aims at finding an (s,t)-path of minimum cost using exactly k distinct colors, for a given $k \in \mathbb{Z}_+$. The SMPP was used to discover light paths in WDM optical networks (Santos et al., 2016), including two extensions of this problem, namely, the Multiple k-Multicolor Paths Problem and the Absolute Multiple k-Multicolor Paths Problem. In the former, a solution is said to be feasible if either each pair of paths is link-disjoint or they use different colors for every shared link, while in the latter, only the first condition is observed. This article proposed two branch-and-bound algorithms, two ILP formulations, and heuristics to efficiently handle these problems.

Lastly, we cite the Colored Path Problem, which aims at finding an (s,t)-path using at most k distinct colors in a vertex-colored graph. It has applications in computational geometry and motion planning (Eiben and Kanj, 2020).

In this work, we deal with the k-Color Shortest Path Problem (k-CSPP), which is NP-Hard (Ferone et al., 2019; Dehouche, 2020). Consider a weighted digraph $D = (V, A)$. Every arc $(i, j) \in A$ has a positive cost d_{ij} and a color $c(i, j)$. Given vertices $s, t \in V$ and a positive integer k, the k-CSPP consists in finding an (s, t)-path of minimum cost while using at most k distinct arc colors. We show an instance of the problem in Fig. 1.

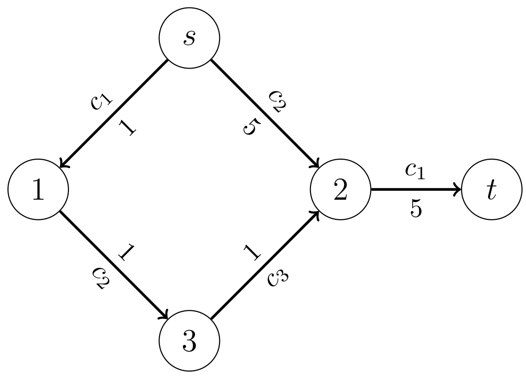

Figure 1: The optimal (s,t)-path for the k-CSPP, $k \geq 3$, is $s \to 1 \to 3 \to 2 \to t$ of cost 8 having colors c_1, c_2, and c_3. For $k = 2$, the optimal solution is $s \to 2 \to t$ of cost 10 having colors c_1 and c_2.

Among existing solution strategies for the k-CSPP, there exists an ILP model, a specialized Branch-and-Bound (B&B) algorithm (Ferone et al., 2019) and a dynamic programming (DP) approach (Ferone et al., 2021). We also find a pseudo-polynomial Dijkstra-based heuristic and an instance reduction procedure (Cerrone and Russo, 2023). Numerical results proved that the solution approaches can solve almost all benchmark instances very efficiently. Nevertheless, B&B and DP find troubles in dealing with new classes of challenging instances generated following the idea proposed by Kumar (2019), while the existing ILP model (Ferone et al., 2019) requires large execution times. This encouraged us to develop new solution techniques to overcome the difficulty in tackling the new instances of the problem.

As the main contributions of this work, we propose new valid inequalities for the k-CSPP, including cuts to be used in a Branch-and-Cut (B&C) algorithm. We also investigate how the value of k impacts the difficulty of instances. Our computational experiments carried out on benchmark instances from the literature show that they can be easily solved after using the reduction procedure (Cerrone and Russo, 2023). Numerical results show the efficiency of the proposed valid inequalities to improve the linear relaxation and the execution time to solve new k-CSPP challenging instances. We also provide numerical results obtained with the heuristic procedure of Cerrone and Russo (2023), the B&B algorithm of Ferone et al. (2019) and the DP algorithm of Ferone et al. (2021). Finally, we discuss situations where the aforementioned heuristic procedure fails to obtain feasible solutions for hard instances.

The remainder of this text is organized as follows. Section 2 presents ILP models for the k-CSPP. Section 3 introduces valid inequalities for the problem. Comments on literature solution approaches are in Section 5. Section 6 reports computational experiments and comparative analysis. Finally, Section 7 presents concluding remarks.

2. Problem formulations

Initially, for every vertex $v \in V$ of $D = (V, A)$, denote the out-neighborhood (resp. in-neighborhood) of v by $N^+(v) = \{i \in V \mid (v, i) \in A\}$ (resp. $N^-(v) = \{i \in V \mid (i, v) \in A\}$). Let C be the set of arc colors of D. The first model (FFP) for the k-CSPP is due to Ferone et al. (2019). Let x_{uv}, for all $(u, v) \in A$, be a binary decision variable to represent whether arc (u, v) belongs to the solution ($x_{uv} = 1$) or not ($x_{uv} = 0$). Furthermore, for all $h \in C$, let y_h be a binary decision variable to determine whether color h is in the solution ($y_h = 1$) or not ($y_h = 0$). The ILP model is as follows.

$$\text{(FFP)} \quad \min \sum_{(u,v)\in A} d_{uv} x_{uv} \tag{1}$$

$$\text{s.t.} \sum_{v\in N^-(u)} x_{vu} - \sum_{v\in N^+(u)} x_{uv} = \begin{cases} -1 & \text{if } u = s \\ +1 & \text{if } u = t \\ \;\;0 & \text{otherwise} \end{cases}, \quad \forall\, u \in V, \tag{2}$$

$$x_{uv} \leq y_{c(u,v)}, \quad \forall\, (u, v) \in A, \tag{3}$$

$$\sum_{h\in C} y_h \leq k, \tag{4}$$

$$x_{uv} \in \{0,1\}, \quad \forall\, (u,v) \in A, \tag{5}$$

$$y_h \in \{0,1\}, \quad \forall\, h \in C. \tag{6}$$

The objective function (1) minimizes the cost of the (s,t)-path. Flow conservation constraints (2) guarantee the path connectivity. Constraints (3) ensure that if arc (u,v) is present in the path, then color $c(u,v)$ is also used. Constraint (4) imposes that at most k distinct colors are in the solution. Finally, constraints (5) and (6) are the domain of the variables. Model (FFP) has $O(|V|+|A|+1)$ constraints and $O(|A|+|C|)$ decision variables.

Observe that if a given color does not belong to the solution, then all variables w.r.t. the arcs of that color can be set to zero. Because the integrality constraints (5) on the x variables, the integrality on y is irrelevant.

Now, we present a new formulation for the k-CSPP that is based on a valid inequality to cut off any infeasible path having more than k distinct arc colors. Initially, note that we can estimate the value of y_h in any (s,t)-path P according to $\frac{1}{|A_h^P|} \sum_{(u,v)\in A_h^P} x_{uv}$, where A_h^P is the set of arcs of P with color h. The set of distinct colors in P, say $C(P)$, must contain at most k elements. Consequently, model (PCM) below is valid for the k-CSPP.

$$\text{(PCM)} \quad \min \sum_{(u,v)\in A} d_{uv} x_{uv}$$

$$\text{s.t. } (2), (5), \text{ and}$$

$$\sum_{h\in C(P)} \frac{1}{|A_h^P|} \sum_{(u,v)\in A_h^P} x_{uv} \leq k, \quad \forall\, P \in \mathcal{P}, \tag{7}$$

where $\mathcal{P}$ stands for the set of all (s,t)-paths of D. When considering the worst-case scenario, where D is complete, for each permutation of vertices there is a corresponding path. Thus, fixed s and t, the number of constraints (7) in model (PCM) is $O((|V|-2)!)$ and of variables is $O(|A|)$. Because of the possible huge number of (s,t)-paths in $\mathcal{P}$, one can explore this formulation as cuts in a B&C scheme.

To illustrate the generation of constraints (7) for model (PCM), consider the digraph

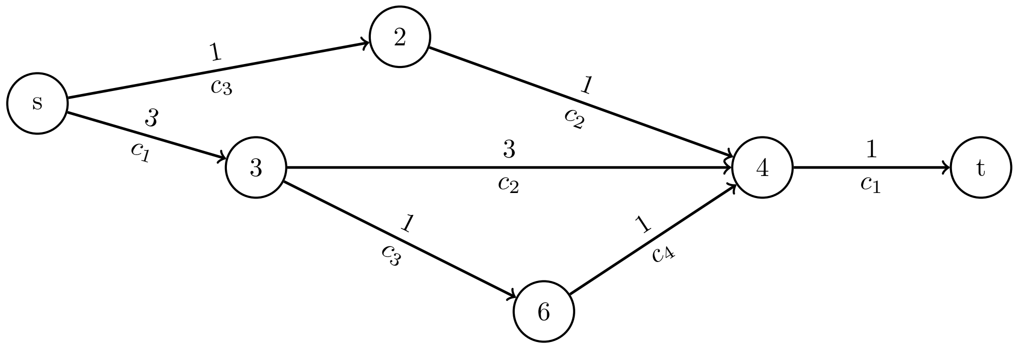

Figure 2: An instance of the k-CSPP for $k = 2$.

depicted in Fig. 2. Let s and t be the source and destination nodes, respectively, and assume $k = 2$. An example of unconstrained (s,t)-path for this digraph is $P_0 = \{(s,2),(2,4),(4,t)\}$ of cost 3 having three distinct colors (c_1, c_2, c_3). For $k = 2$, P_0 is infeasible. We deduce that at least one color within P_0 must remain unused in the optimal solution for the 2-CSPP. To address this, we estimate the occurrence of any color appearing in P_0 by dividing the sum of the arc decision variables of that color by the number of arcs sharing the same color in the path. This gives the first valid cut $x_{s,2} + x_{2,4} + x_{4,t} \leq 2$. Assume that upon adding this cut into the model and reoptimizing it, we obtain a new infeasible path $P_1 = \{(s,3),(3,6),(6,4),(4,t)\}$ of cost 6. It has three colors (c_1, c_3, c_4). When observing that now we have two arcs $(s,3)$ and $(4,t)$ of color c_1, we obtain the second valid cut $\frac{1}{2}(x_{s,3} + x_{4,t}) + x_{3,6} + x_{6,4} \leq 2$. After reoptimizing the model with these two cuts, we reach the optimal solution $P^* = \{(s,3),(3,4),(4,t)\}$, which has cost 7 and contains two colors (c_1, c_2).

3. Valid Inequalities

In this section, we propose valid inequalities for the k-CSPP. Initially, we know that in any (s,t)-path, at most one arc leaves or enters vertex $u \in V$. Consequently,

$$\sum_{v \in N^+(u)} x_{uv} \leq 1, \quad \forall\, u \in V \setminus \{t\}, \tag{8}$$

$$\sum_{v \in N^-(u)} x_{vu} \leq 1, \quad \forall\, u \in V \setminus \{s\}. \tag{9}$$

The second set of valid inequalities concerns arcs of same color leaving or entering a given vertex $u \in V$. Let C_u^+ (resp. C_u^-) be the set of colors in the out-neighborhood (resp. in-neighborhood) of vertex u; and $\bar{R}(u)$ be the set of vertices not reachable by u in D. Also, denote by $N_h^+(u)$ (resp. $N_h^-(u)$) the set of end-nodes of arcs of color h that leave (resp. enter) u.

Proposition 1. *In any (s,t)-path, the number of arcs of color h leaving (or entering) vertex u is limited above by y_h.*

$$\sum_{v \in N_h^+(u)} x_{uv} \leq y_h, \quad \forall\, u \in V,\ \forall\, h \in C_u^+, \tag{10}$$

$$\sum_{v \in N_h^-(u)} x_{vu} \leq y_h, \quad \forall\, u \in V,\ \forall\, h \in C_u^-. \tag{11}$$

Proof. Straightforward. Note that both (10) and (11) cut off fractional solutions where two or more arcs of the same color leave or enter a given node. They strengthen some constraints (3). We can extend the idea of Proposition 1 for pairs of non-reachable vertices. This situation appears in fractional linear relaxed solutions where the digraph induced by x variables contains sub-paths between s and t presenting vertices u and v, with $v \in \bar{R}(u)$ and $u \in \bar{R}(v)$. □

Proposition 2. *Consider vertices $u, v \in V$ such that $v \in \bar{R}(u)$ and $u \in \bar{R}(v)$. The number of arcs of color h entering or leaving u and v is limited above by y_h.*

$$\sum_{j \in N_h^+(v)} x_{vj} + \sum_{i \in N_h^+(u)} x_{ui} \leq y_h, \quad \forall\, u, v \in V,\ v \in \bar{R}(u),\ u \in \bar{R}(v),\ \forall h \in C, \tag{12}$$

$$\sum_{j \in N_h^+(v)} x_{vj} + \sum_{i \in N_h^-(u)} x_{iu} \leq y_h, \quad \forall\, u, v \in V,\ v \in \bar{R}(u),\ u \in \bar{R}(v),\ \forall h \in C, \tag{13}$$

$$\sum_{j \in N_h^-(v)} x_{jv} + \sum_{i \in N_h^-(u)} x_{iu} \leq y_h, \quad \forall\, u, v \in V,\ v \in \bar{R}(u),\ u \in \bar{R}(v),\ \forall h \in C. \tag{14}$$

Proof. Consider two vertices, u and v, within the vertex set V, such that $v \in \bar{R}(u)$ and $u \in \bar{R}(v)$. Since u and v cannot be mutually reachable, they cannot simultaneously appear in a feasible solution. Therefore, if color h is part of the solution, i.e., $y_h = 1$, then in the following scenarios, at most one arc of color h can be included in the solution: (i) departing from both v and u, resulting in inequality (12); (ii) departing from one of these nodes (e.g., v) and entering the other one (e.g., u), leading to inequality (13); and (iii) entering both v and u, leading to inequality (14). □

In fact, the idea of Proposition 2 applies to any set $S \subseteq V$ containing only non-reachable vertices.

Proposition 3. *Let $S \subseteq V$ be such that for any two vertices u and v of S, $v \in \bar{R}(u)$ and $u \in \bar{R}(v)$. Let $Q \subseteq S$. The number of arcs of color h leaving Q and entering $S \setminus Q$ is limited above by y_h if color h belongs to the solution.*

$$\sum_{u \in Q} \sum_{j \in N_h^+(u)} x_{uj} + \sum_{v \in S \setminus Q} \sum_{j \in N_h^-(v)} x_{jv} \leq y_h, \quad \forall\, S \subseteq V,\ \forall\, Q \subseteq S,\ \forall\, h \in C. \tag{15}$$

Proof. The result follows from the fact that at most one of the arcs of a given color h incident to the non-reachable vertices of S can belong to the solution if this color also belongs. □

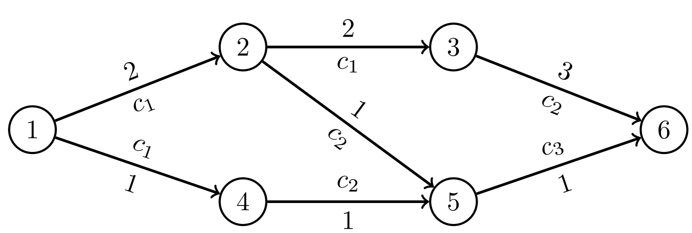

Figure 3: An arc-colored digraph.

To explain Propositions 2 and 3, we examine the digraph depicted in Fig. 3. Note, w.r.t. nodes 2 and 4, that $2 \in \bar{R}(4)$ and $4 \in \bar{R}(2)$. Arc $(1,4)$ of color c_1 enters node 4 and the one $(2,3)$ of the same color leaves node 2. Thus, we can generate the corresponding inequality (13) from Proposition 2: $x_{1,4} + x_{2,3} \leq y_{c_1}$. Now, consider nodes 3 and 5, where $3 \in \bar{R}(4)$ and $5 \in \bar{R}(3)$. Let $S = \{3,5\}$ and take $Q = \{3\} \subset S$. We have two arcs $(2,5)$ and $(4,5)$ of color c_2 entering node 5, which belongs to $\{3,5\} \setminus \{3\}$. There exists one arc

$(3,6)$ of the same color leaving $S = \{3,5\}$. Thus, we can also generate the following valid inequality (15) from Proposition 3: $x_{2,5} + x_{4,5} + x_{3,6} \leq y_{c_2}$.

The next proposition explores the fact that if there exists an arc (u,v) where u is not reachable by v and vertices $j \in N^+(u) \setminus \{v\}$ that do not reach v, then only one of the arcs of the same color, say h, leaving u and not reaching v as well as from v to $N^+(v)$ can belong to the solution. For this, we introduce new notations. Let $\bar{I}(u,v) = \{j \in N^+(u) \setminus \{v\} \mid v \in \bar{R}(j)\}$ denotes the set of vertices in the out-neighborhood of u that do not reach v; and $L = \{(u,v) \in A \mid u \in \bar{R}(v),\ \bar{I}(u,v) \neq \emptyset\}$ be a non-empty set of arcs such that their tails u are not reached by their heads v and there is at least one vertex in the out-neighborhood of u that does not reach v.

Proposition 4. *For every arc $(u,v) \in L \neq \emptyset$, the number of arcs of color $h \in C_v^+$ from u to vertices in $\bar{I}(u,v)$ and leaving v of that color is limited above by y_h.*

$$\sum_{\substack{j \in \bar{I}(u,v) \\ c(u,j)=h}} x_{uj} + \sum_{z \in N_h^+(v)} x_{vz} \leq y_h, \quad \forall\ (u,v) \in L,\ h \in C_v^+. \tag{16}$$

Proof. Consider an arc $(u,v) \in L$. By assumption, $u \in \bar{R}(v)$. For all $j \in \bar{I}(u,v)$, with $v \in \bar{R}(j)$, the arcs (u,j) of any color $h \in C_v^+$ cannot be used in the same solution with arcs leaving v of this color. Consequently, the sum of the corresponding x variables for these arcs is limited above by y_h. □

We give an example of Proposition 4 for the digraph on the right side of Fig. 4. Arcs $(s,1)$, $(1,4)$, $(2,3)$, $(2,5)$ and $(4,2)$ have non-empty sets $\bar{I}(s,1) = \{2\}$, $\bar{I}(1,4) = \{3\}$, $\bar{I}(2,3) = \{5\}$, $\bar{I}(2,5) = \{3\}$ and $\bar{I}(4,2) = \{t\}$. Hence, $L = \{(s,1),(1,4),(2,3),(2,5),(4,2)\}$. Thus, for the arc $(2,3) \in L$ of color c_1, we obtain the valid inequality $x_{2,5} + x_{3,t} \leq y_{c_1}$.

We now extend the idea behind Proposition 4. Let us denote by $W(v) = \{(u,j) \in A \mid u \in N^-(v),\ u \in \bar{R}(v),\ j \in \bar{I}(u,v)\}$ such that if (u',j') and (u'',j'') are in $W(v)$, then $j' \in \bar{R}(j'')$ and $j'' \in \bar{R}(j')$. Here, $W(v)$ is the maximal non-empty set of arcs leaving the in-neighborhood of $v \in V \setminus \{s,t\}$ to vertices that do not reach v and satisfy the property

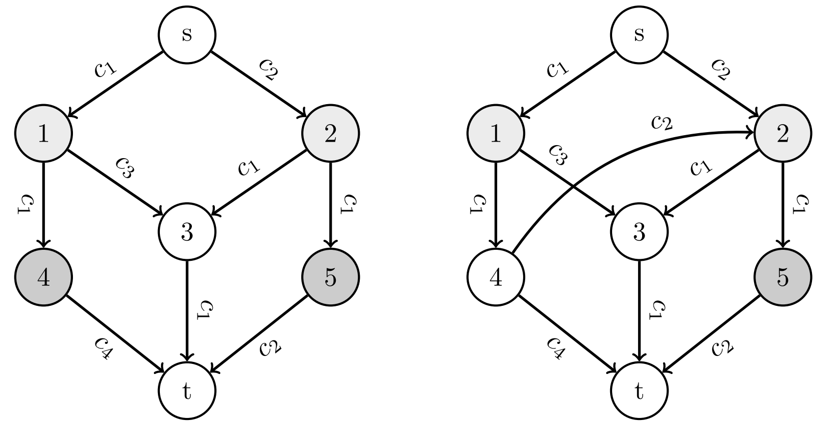

Figure 4: Two arc-colored digraphs. The right one corresponds to the left digraph with the addition of the arc $(4, 2)$.

that the head of every arc in this set is not reached by v or by the head of any other arc in $W(v)$. Also, denote the set of arcs of color h in $W(v)$ by $W_h(v)$.

Proposition 5. *If $W(v) \neq \emptyset$, for some vertices $v \in V \setminus \{s, t\}$, then the number of arcs of a color $h \in C_v^+$ in $W_h(v)$ and those leaving v of this color is at most y_h.*

$$\sum_{(u,j) \in W_h(v)} x_{uj} + \sum_{j \in N_h^+(v)} x_{vj} \leq y_h, \quad \forall\, v \in V \setminus \{s, t\},\ \forall\, h \in C_v^+,\ W_h(v) \neq \emptyset. \tag{17}$$

Proof. Consider a vertex $v \in V \setminus \{s, t\}$ and a color $h \in C_v^+$ for which $W_h(v) \neq \emptyset$. By definition of $W(v)$, at most one arc of this set can belong to the solution because their heads do not reach each other. The same is valid for the arcs leaving v. Moreover, neither these heads reach v nor v reaches them. Consequently, at most one of all these arcs can be in the solution and, in particular, the ones of color h if it is in the solution. Thus, the result follows. □

We show an example of Proposition 5 for the digraph on the left side of Fig. 4. For instance, let $v = 3$ and consider color c_1 in C_3^+ of the arc $(3, t)$. We have $N^-(3) = \{1, 2\}$, $\bar{I}(1, 3) = \{4\}$ and $\bar{I}(2, 3) = \{5\}$. Hence, $W(3) = W_{c_1}(3) = \{(1, 4), (2, 5)\}$. Therefore, $x_{1,4} + x_{2,5} + x_{3,t} \leq y_{c_1}$ is a valid inequality for this instance.

The next valid inequality is based on the fact that, if two arcs (u, w) and (v, r) of the same color belong to the solution, then there must also be a (w, v)-path or a (r, u)-path.

Proposition 6. *If two non-consecutive arcs (u, w) and (v, r) of the same color h belong to the solution, then their colors also belong, and we must have at least one arc between them.*

$$x_{uw} + x_{vr} \leq \sum_{\substack{i \in N^+(w) \\ v \notin \bar{R}(i)}} x_{wi} + \sum_{\substack{i \in N^+(r) \\ u \notin \bar{R}(i)}} x_{ri} + y_h, \quad \forall\, h \in C,\ \forall\, (u, w), (v, r) \in A_h. \tag{18}$$

Proof. If two non-consecutive arcs of color h, say $(u, w), (v, r) \in A_h$, belong to the solution, then color h must also belong. Moreover, one arc must leave w and later reach v or one arc must leave r and later reach u. Thus, the result follows. □

If we consider an (s, t)-cut $[S, V \setminus S]$ of D such that $s \in S$, $t \in V \setminus S$, and $[V \setminus S, S] = \emptyset$, then the sum of the arcs of color h in $[S, V \setminus S]$ is limited above by y_h according to

$$\sum_{\substack{(u,v) \in [S, V \setminus S] \\ c(u,v) = h}} x_{uv} \leq y_h, \quad \forall\, S \subseteq V \mid s \in S, t \in V \setminus S,\ \forall\, h \in C[S, V \setminus S] \tag{19}$$

4. Separation procedures for some proposed families of valid inequalities

In this section, we describe separation procedures for the valid inequalities proposed in Section 3. In the following, to verify whether $u \in \bar{R}(v)$ for a given pair of vertices u and v, we construct a $|V| \times |V|$ binary matrix T, where $T_{u,v} = 1$ if u is reached by v, by running a breadth-first search (BFS) starting from every vertex of the digraph. We employ heuristic procedures to obtain maximal sets of non-reachable nodes, as explained below.

We derive inequalities (15) by initiating with a singleton set $S = \{v\}$ comprising pairwise unreachable vertices, one for each $v \in V \setminus \{s, t\}$. For all $u \in V \setminus \{s, t, v\}$ with label greater than v, we include u in S if it is neither reachable by nor connected to any vertex already present in S. Subsequently, we derive an inequality (15) for each subset $Q \subseteq S$, where $|Q|$ takes on values in the set $\{0, 1, |S| - 1, |S|\}$.

We generate inequalities (18) using the CPLEX user cut callback, which is based on the support graph associated with non-null arc variables x concerning a CPLEX B&B node solution. For each pair of arcs (u, w) and (v, r) of the same color h, within any B&B node solution that violates the corresponding inequality for that pair, we derive an inequality (18). Notably, if, for this pair of arcs, we observe that $v \notin \bar{R}(i)$ holds for all $i \in N^+(w)$, we exclude the corresponding inequality (18) because we can demonstrate that it is weaker than a "modified combination" of constraints (2) for node w.

Finally, to obtain inequalities (19), we first describe a randomized algorithm that computes the minimum cut of a digraph (Motwani and Raghavan, 1995). The idea behind the algorithm is to contract arcs randomly chosen until a unique arc remains. The label of the resulting vertex of an arc contraction is the union of the set of labels of the arc extremities. Only one arc is considered in cases of duplicated arcs or arcs in opposite directions between a given pair of nodes. The algorithm returns the partition of vertices V_1 and V_2 given by the extremities of the unique remaining arc. Given such partition, we check whether the cut of arcs $[V_2, V_1]$ is empty. If so, we add the corresponding inequality (19) to model (FFP). Otherwise, while $[V_2, V_1] \neq \emptyset$, for every arc (u, v) in this cut, we move u from V_2 to V_1. The resulting partition is then used to generate that inequality. For each instance, we run the routine 60 times to obtain 60 inequalities of this type.

5. Solution strategies from the literature

In this section, we present existing solution strategies for the k-CSPP.

5.1. The heuristic procedure of Cerrone and Russo (2023)

Cerrone and Russo (2023) devised a constructive pseudo-polynomial algorithm named Color Constrained Dijkstra Algorithm (CCDA), which is rooted in Dijkstra's algorithm. The underlying principle of CCDA is the iterative construction of an (s, t)-path. In doing so, it takes into account penalties from a predefined list of values Λ and applies them to arc costs when their colors have not yet been incorporated into the evolving path.

Let $w_{\min}$, w_{mean}, and $w_{\max}$ represent the minimum, average, and maximum arc costs within set A. The values defining penalties in Λ are as follows: Λ are $\{0,\ w_{\min}/4,\ w_{\min}/2,\ w_{\min},$

$2\times w_{\min}$, $w_{\text{mean}}/4$, $w_{\text{mean}}/2$, w_{mean}, $w_{\max}\}$. The modified Dijkstra's algorithm, with penalties, operates in a conventional manner except for how it calculates arc costs.

In ascending order, CCDA sequentially considers penalty values $\lambda \in \Lambda$, one at a time, starting from the smallest and progressing to the largest. When updating the cost estimate of the path from source s to a vertex $v \in V \setminus \{s\}$, if the color of arc (u, v) is absent in the partial (s, v)-path, then the algorithm considers the penalized arc cost $d_{uv} + \lambda$. It is necessary to keep the predecessor of each vertex as well as the list of colors in the (s, v)-path. The algorithm is intended to stop with a feasible path when it reaches t.

The complexity of CCDA depends on that of Dijkstra's algorithm and the number of penalty values. A similar idea is adopted in the DP algorithm detailed hereafter.

It is worth noting that we have observed situations in which CCDA terminates without yielding a feasible solution. The effectiveness of this heuristic hinges on the range of penalty values defined within Λ. Indeed, in Fig. 5, if we define $\Lambda \subset [0, 3]$ as Cerrone and Russo (2023) suggest, then for every $\lambda \in \Lambda$, CCDA always returns the path $s \to 3 \to 4 \to t$ of three colors. This becomes problematic when $k = 2$. To overcome this drawback, we allow the maximum value of λ to exceed the highest arc cost in the digraph, e.g. $2 \times w_{\max}$. In the example from Fig. 5, with $\lambda = 6$, CCDA successfully identifies the optimal two-color path $s \to 1 \to 2 \to t$.

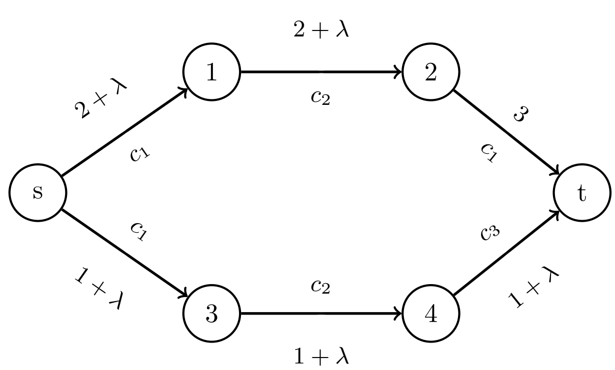

Figure 5: Digraph instance for which the CCDA heuristic fails obtaining a feasible (s, t)-path with at most $k = 2$ colors if we adopt penalties from the interval $\lambda \in [0, 3]$. The notation on each arc (u, v) is $d_{uv} + \lambda$, except for the arc $(2, t)$ of cost 3 because its color is the same as the one of the arc $(s, 1)$.

5.2. The Branch-and-Bound procedure of Ferone et al. (2019)

The approach used in the B&B procedure by Ferone et al. (2019) for the k-CSPP involves solving a shortest path problem for each node in the search tree. This is achieved by recognizing that relaxing constraints (3) and (4) transforms the model (FFP) into a standard shortest path formulation. When a node solution utilizes more than k colors, one refrains from using arcs associated with some colors within the solution and compute the shortest path within the resulting subgraph, excluding the forbidden colors.

Let z_i denote the solution of the i-th B&B node generated in the search tree. If $|C(z_i)| > k$, we divide this node into $|C(z_i)|$ subproblems, one for each color present in z_i that is required not to be part of the solution. The order of generation of these nodes is determined by the absolute frequency of colors in z_i, with the less frequent colors being removed first. One updates the incumbent solution whenever a feasible path with a better solution value is discovered. We alert that this "B&B procedure" possibly generates non-disjoint regions of feasible solutions for the subproblems in the search tree.

5.3. The dynamic programming algorithm of Ferone et al. (2021)

The label-setting DP algorithm of Ferone et al. (2021) relies on the concepts of feasible and dominated paths. Let $P_{s,u}$ denote an (s,u)-path. Define $P(u)$ as the set of all paths from s to u. A path $P_{s,u}$ is said to be feasible if $|C(P_{s,u})| \leq k$. Given two paths $P'_{s,u}$ and $P''_{s,u}$, path $P'_{s,u}$ is said to dominate $P''_{s,u}$ if $C(P'_{s,u}) \subseteq C(P''_{s,u})$, $d(P'_{s,u}) \leq d(P''_{s,u})$, and at least one of these conditions is strict. Dominance conditions avoid exploring unfruitful paths.

The algorithm retains a record of all feasible and non-dominated paths, denoted as $P_{s,u}$, for each node u within the digraph throughout its execution. Additionally, it manages a queue of paths currently under construction, which may expand arbitrarily. Given the potential volume of paths stored in memory, a good label extraction policy is necessary.

Five extraction rules were proposed. The first one is a Dijkstra-like rule (DR), where the next path to be explored is the one of minimum cost. A standard First-In First-Out (FIFO) rule – where the extracted path is the one in the queue for the longest time – and a Last-In First-Out (LIFO) rule – where we extract the last path inserted in the queue – were also explored. The fourth rule is the Small-Label-First (SMF). In this rule, every time a new path P is to be added to the list of paths to a node, one checks if $d(P) < d(P')$, where P' is the path currently at the head of the list. If the condition is satisfied, P is placed at the head of the list, otherwise it is placed at its tail. The last rule is the A^* one, which selects the path P that minimizes $d(P) + \pi(last(P), t)$, where $\pi(last(P), t)$ represents the shortest path value between the last visited node in P, denoted as $last(P)$, and the target node t.

6. Computational results

In this section, we present numerical experiments performed on a PC Intel Core i7-3770, 8×3.40 GHz, 16 GB DDR3 RAM with Ubuntu 20.4 LTS 64 bits. We use Julia 1.8.5 with JuMP package to implement models for CPLEX 22.1 configured with one thread. The time limit for each instance is set to 1800 seconds.

We adopt benchmark instances (grid and random digraphs) from the literature (Ferone et al., 2019) and generate novel classes (groups) of layered-based digraphs, as similar instances showed to be hard to handle for the MCPP (Kumar, 2019). Each layered digraph is composed of w layers of r vertices per layer, in addition to source s and destination t vertices. The source s connects to every vertex of the first layer, while all the vertices in the last layer connect to the destination t. In a standard layered digraph, there is an arc from every vertex of layer i to every vertex of layer $i + 1$, with $i := 1, \cdots, w - 1$.

We generate 60 new instances (available under request) for the problem, divided into three groups of 20 instances. All the new test-bed digraphs have $2 + 15 \times 10$ vertices: a source, and a destination, as well as $w = 15$ columns (layers) with $r = 10$ vertices each one. The groups are categorized as follows:

- Group 1 contains standard layered digraphs with $wr + 2$ vertices and $2r + (w - 1)r^2 = 1420$ arcs. We uniformly choose their arc costs from the integer interval $[1, 1000]$;

- Group 2 is composed of modified layered digraphs. We first generate a digraph as those of Group 1. Then, we create an arbitrary number (from the integer interval $[10, 30]$) of jump-arcs. To obtain a jump-arc (u, v), we randomly choose u and v from non-neighbor layers L_i and $L_{i'}$, respectively, with $i < i'$. We uniformly choose the cost of a jump-arc from the integer interval $[d_{\max}, d_{\max} + 30000]$, where $d_{\max}$ is the highest arc cost among non-jump-arcs. The idea behind using a few jump-arcs with high costs is to allow the heuristics to easily find a feasible (possibly costly) path;

- Group 3 has digraphs as those of Group 1, but initially with an empty set of arcs. For each layer L_i, $i := 1, \cdots, w-1$, we select at random 3 distinct vertices and form a directed clique Q_i with them. Then, for each pair of vertices (u, v), where $u \in Q_i$ and $v \in L_i - Q_i$, we create arcs (u, v) and (v, u) with probability $p = 1/2$. Finally, we add an arc from every vertex of layer L_i, except from the vertex with the smaller label of Q_i, to every vertex of layer L_{i+1}. All arc costs of this group are chosen from the integer interval $[1, 1000]$. This digraph topology aims to allow any (s,t)-path to pass through an internal arc of the layers. The number of arcs in any feasible path for this group can be at most the number of layers more than those for the instances of Group 1. The likelihood of a path utilizing an internal arc within a layer is not insignificant. To illustrate this, let us begin by recalling that in a clique Q_i within a layer L_i, the vertex with the smaller label lacks an arc connecting it to the subsequent layer, L_{i+1}. If the path accesses a layer through a vertex that does not belong to its clique, it will not traverse an internal arc within that layer. However, if the path utilizes one of the three vertices in Q_i, we encounter two distinct scenarios. In the first scenario, if the path reaches the vertex with the smaller label in Q_i, the likelihood of utilizing an internal arc corresponds to the probability of selecting that particular vertex from among the r vertices within the layer, i.e., $\frac{1}{r}$. In the second case, when the path uses one of the two remaining vertices within the clique, each of these vertices is internally connected to the other two vertices within Q_i and approximately half of the remaining $r-3$ nodes within the same layer. This is because we adopt a probability of $p = 1/2$ to establish an internal arc in each layer. Additionally, these vertices are externally connected to r vertices in the subsequent layer L_{i+1}, where $i < w$. Consequently, the probability of employing an internal arc in this scenario is $2(\frac{2+\frac{r-3}{2}}{r+2+\frac{r-3}{2}})$. Therefore, the cumulative likelihood of a path utilizing an internal arc within a layer, excluding the final layer, is calculated as $\frac{5r+3}{r(3r+1)}$. Since paths within this group can have (and explore) a greater number of arcs compared to those in Groups 1 and 2, this particular set of instances is designed to be the most challenging for the problem.

Arc colors of our new instances are chosen based on a uniform distribution over the integer interval $[1, |C|]$, where we set $|C| = \lfloor |A|/4 \rfloor$. Finally, we define k as the minimum number of colors allowing an (s,t)-path, obtained after solving the MCPP for each instance individually. For instances of Group 2, we evaluate the MCPP in the digraph without jump-arcs. Both choices for $|C|$ and k are based on an extensive set of preliminary tuning experiments. Given the shortest (s,t)-path having k' colors, for values of $k \geq k'$ the k-CSPP is easy. To give an idea of the problem difficulty (in terms of cpu execution time) for values of $k < k'$ and distinct values of $|C|/|A|$, we depict some related experiments in Fig. 6 for four layered digraphs obtained as those of Group 1 with the same number of vertices and arcs. The axes in Fig. 6 represent the color density $|C|/|A|$, the maximum number of colors k in the solution path, and the execution time in seconds *cpu*. We observe that the instances require higher computational time when $|C|/|A| \in (0.25, 0.55]$ and the k values are equal to the minimum number of colors related to the MCPP solutions for these digraphs.

Tables 1 and 2 provide information regarding the characteristics of the new instances from Groups 1, 2, and 3, as well as the random and grid digraphs (Ferone et al., 2019). I these tables, we identify each instance with a label *inst*. They receive labels from 1 to 60 in Table 1. In Table 2, the instance identifier corresponds to the original instance name (Ferone et al., 2019). In Table 1, all instances have $|V| = 152$ vertices, while the number of arcs $|A|$ varies across different groups. The limit on the number of colors is k, and the known optimal solution value is *opt*. To enhance efficiency, we employ a graph reduction algorithm, which eliminates arcs proven to be unnecessary for the optimal solution based on feasible solutions obtained through the CCDA heuristic (Cerrone and Russo, 2023). We use the heuristic solution in the CPLEX solver as a cutoff value. For the instances where we have a reduction on their size, $R(V)$ and $R(A)$ denote, respectively, the reduced set of vertices

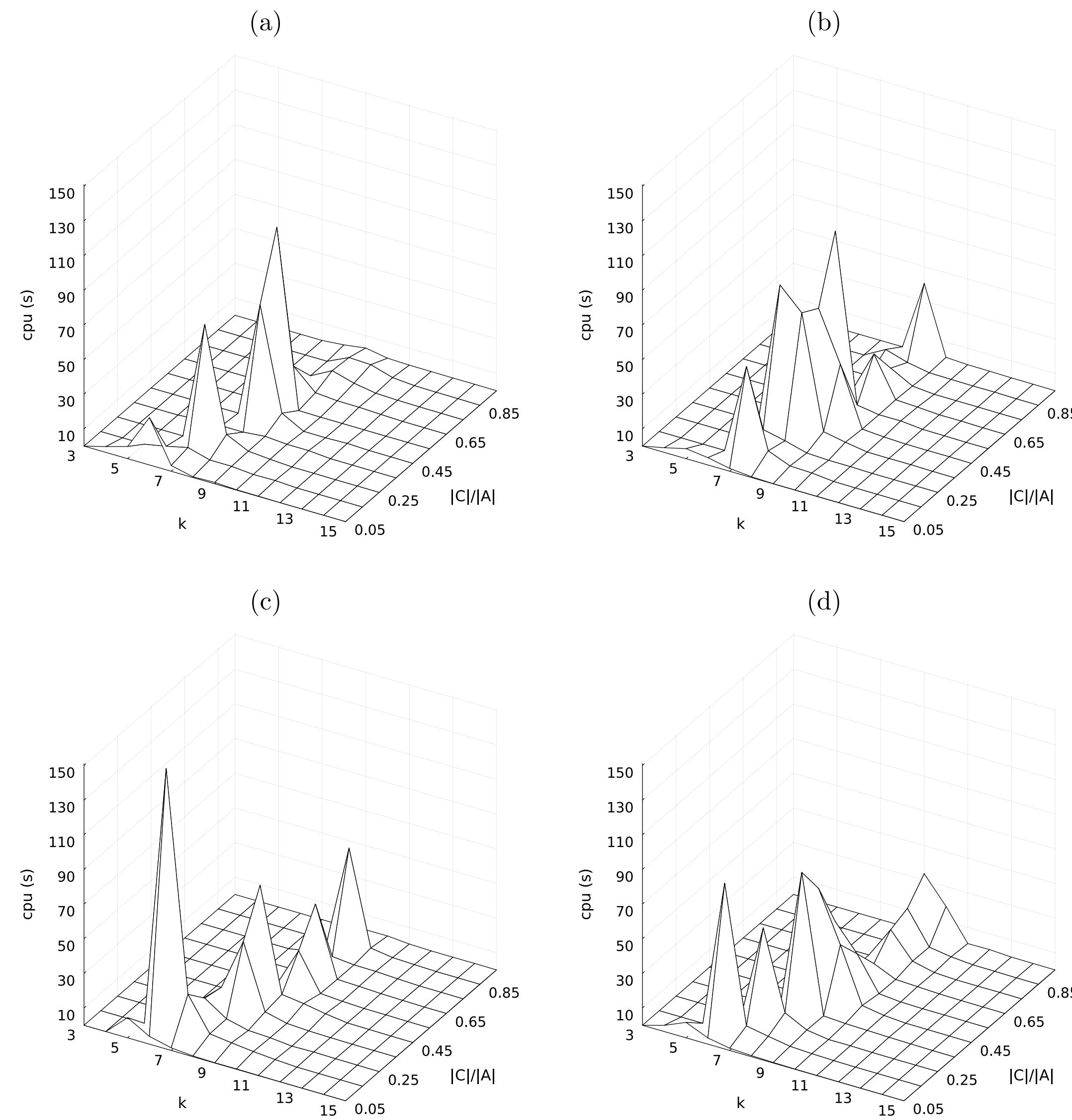

Figure 6: Variation of execution time in function of $|C|/|A|$ and k for 4 layered digraphs.

and arcs. We have to mention that the reduction algorithm was not able to remove any arc or vertex of the instances of Groups 1, 2, and 3. In contrast, in line with findings reported by Cerrone and Russo (2023), we verify a drastic reduction on the number of vertices and arcs for the benchmark instances of Ferone et al. (2019).

Table 1: Details about the instances of Groups 1, 2, and 3.

Group 1				Group 2				Group 3			
inst	$\|A\|$	k	*opt*	*inst*	$\|A\|$	k	*opt*	*inst*	$\|A\|$	k	*opt*
1	1420	6	9242	21	1430	7	6470	41	1704	7	7909
2	1420	7	6358	22	1447	7	6211	42	1722	7	8503
3	1420	7	5953	23	1439	7	5897	43	1692	7	7400
4	1420	7	5056	24	1435	7	5602	44	1698	7	7838
5	1420	7	6495	25	1444	7	7288	45	1720	7	7736
6	1420	7	5382	26	1431	7	6824	46	1672	8	6293
7	1420	7	6774	27	1434	7	6969	47	1676	8	4963
8	1420	7	6345	28	1434	7	6956	48	1692	8	5458
9	1420	7	4086	29	1430	7	5186	49	1690	8	4447
10	1420	7	7052	30	1445	7	4763	50	1700	8	5095
11	1420	7	6128	31	1445	7	6550	51	1652	7	6737
12	1420	7	6097	32	1435	7	7029	52	1692	8	4636
13	1420	7	5541	33	1438	7	5600	53	1688	8	5596
14	1420	7	4981	34	1442	7	6102	54	1686	8	4632
15	1420	7	6572	35	1440	7	7624	55	1680	8	5335
16	1420	7	5434	36	1441	7	5124	56	1680	8	5222
17	1420	7	7386	37	1449	7	6778	57	1660	8	4852
18	1420	7	5102	38	1438	7	5761	58	1704	8	4785
19	1420	7	5885	39	1435	7	5006	59	1662	8	6132
20	1420	7	5793	40	1440	7	6845	60	1716	8	5974

Table 2: Details about the benchmarks instances for random and grid digraphs (Ferone et al., 2019).

Random							Grid						
inst	$\|V\|$	$\|A\|$	$\|R(V)\|$	$\|R(A)\|$	k	*opt*	*inst*	$\|V\|$	$\|A\|$	$\|R(V)\|$	$\|R(A)\|$	k	*opt*
R1-27190	75000	750000	16	16	8	242	G1-27000	10000	39600	627	1648	195	6131
R1-27191	75000	750000	35	39	6	201	G1-27001*	10000	39600	209	424	197	6233
R1-27195*	75000	750000	13	13	6	152	G1-27002	10000	39600	402	956	191	6336
R1-27197	75000	750000	12	12	6	253	G1-27003	10000	39600	423	922	196	6200
R1-27199	75000	750000	22321	99827	5	333	G1-27004	10000	39600	256	580	195	6375
R1-27200	75000	750000	120	147	6	236	G1-27005	10000	39600	248	578	197	6079
R1-27202	75000	750000	80	92	8	253	G1-27006	10000	39600	281	624	193	6109
R1-27203	75000	750000	17	20	5	255	G1-27007	10000	39600	220	462	198	6197
R1-27204	75000	750000	24808	119208	5	401	G1-27008	10000	39600	214	442	191	6193
R1-27205	75000	750000	33203	190490	6	426	G1-27009	10000	39600	247	540	196	6181
R2-27001	75000	750000	80	93	6	289	G3-27000	20000	79400	380	812	312	9808
R2-27004	75000	750000	27	29	6	246	G3-27001	20000	79400	514	1154	294	9786
R2-27005	75000	750000	23	25	6	198	G3-27002	20000	79400	489	1126	291	9652
R2-27007	75000	750000	33	38	6	231	G3-27003	20000	79400	323	672	305	9448
R2-27008	75000	750000	15	15	7	196	G3-27004	20000	79400	1145	2884	295	10149
R2-27010	75000	750000	148	180	5	246	G3-27005	20000	79400	437	1040	296	9793
R2-27012	75000	750000	26	28	7	245	G3-27006	20000	79400	688	1520	299	9654
R2-27015	75000	750000	56	65	6	238	G3-27007	20000	79400	319	664	298	9535
R2-27018	75000	750000	61	74	6	219	G3-27008	20000	79400	371	818	295	9455
-	-	-	-	-	-	-	G3-27009	20000	79400	2013	6654	296	9424

$*$ The shortest path solution is feasible for the k-CSPP.

6.1. *Computational results for CCDA, DP, B&B, and the B&C algorithms*

In Table 3, we report numerical results for the CCDA heuristic (Cerrone and Russo, 2023), the Dynamic Programming (DP) procedure (Ferone et al., 2021), the Branch-and-Bound (B&B) algorithm (Ferone et al., 2019), and for our Branch-and-Cut (B&C) based formulation (PCM). The legend includes the instance identifier *inst*, the heuristic solution *ub*, the number of *colors* in the path, and the execution time *cpu* (in seconds) for each solution approach. The time limit for CCDA was 10 seconds, and for DP, B&B, and (PCM) it was 1800 seconds. The number of generated nodes in the B&B (resp. CPLEX B&C) tree is denoted by *bb* (resp. *bc*). For model (PCM), we also report the number of generated *cuts* (7). We separate results for each group of instances by horizontal lines. In the last four lines of each group, we report statistic values of average, median, maximum, and minimum values of each column. The last lines of this table refer to the overall related statistics.

Table 3: Numerical results for CCDA, DP, B&B, and the mathematical formulation (PCM) for the benchmark grid and random digraphs and the instances of Groups 1 and 2.

	CCDA			**DP**	**B&B**		**PCM**		
Instance	*ub*	*cpu*	*colors*	*cpu*	*bb*	*cpu*	*bc*	*cuts*	*cpu*
R1-27190	242	1.6	6	0.8	10	63.2	0	1	1.0
R1-27191	201	2.1	6	0.5	12090	1800.1	4	10	1.5
R1-27195	152	1.1	6	0.5	0	61.0	0	0	0.1
R1-27197	253	1.2	5	0.5	8	64.2	0	1	0.1
R1-27199	333	5.1	5	1.5	10103	1800.2	6508	2439	1800.0
R1-27200	236	5.1	6	0.5	10386	1800.4	39	30	3.7
R1-27202	261	4.4	7	0.5	49	120.8	12	12	1.9
R1-27203	255	4.4	5	0.5	17	83.6	0	2	0.2
R1-27204	401	5.5	5	1.4	11719	1800.5	4737	1759	1800.0
R1-27205	426	6.1	6	3.8	11361	1800.8	3730	2008	1800.0
R2-27001	289	5.0	6	0.5	8	74.2	6	13	3.7
R2-27004	246	2.0	6	0.5	10275	1800.3	0	2	0.3
R2-27005	198	2.6	5	0.6	9882	1801.2	0	8	1.1
R2-27007	231	3.1	6	0.5	365	523.3	8	6	0.7
R2-27008	196	2.6	6	0.5	9	75.9	0	1	0.2
R2-27010	246	5.0	5	0.5	11219	1800.0	48	32	3.2
R2-27012	245	1.3	6	0.5	81	174.3	0	2	0.3
R2-27015	238	3.3	6	0.5	9176	1800.7	11	13	2.2
R2-27018	219	3.4	5	0.5	10404	1800.7	22	17	2.4
Average	256.2	3.4	5.7	0.8	5640.1	1012.9	796.1	334.5	285.4
Median	245.0	3.3	6.0	0.5	9176.0	1800.0	6.0	10.0	1.5
Max	426.0	6.1	7.0	3.8	12090.0	1801.2	6508.0	2439.0	1800.0
Min	152.0	1.1	5.0	0.5	0.0	61.0	0.0	0.0	0.1
G1-27000	6150	0.3	194	5.2	3197	250.5	2803	2296	26.5
G1-27001	6233	0.1	197	0.0	1810	74.3	3	10	0.4
G1-27002	6336	0.2	189	5.4	193	9.4	2170	674	7.3
G1-27003	6203	0.2	194	0.3	198	16.4	262	120	0.6
G1-27004	6375	0.2	195	0.6	601059	1800.0	297	150	1.3
G1-27005	6079	0.2	197	2.8	2195	136.7	665	401	2.3
G1-27006	6109	0.2	193	0.7	11956	856.2	327	174	0.8
G1-27007	6197	0.2	198	0.1	1000	76.5	8	9	0.1
G1-27008	6193	0.1	191	0.0	4456005	1800.1	0	4	0.0

G1-27009	6183	0.1	195	0.1	395	15.9	35	15	0.1
G3-27000	9808	0.3	311	3.8	2507	224.1	166	121	0.7
G3-27001	9792	0.5	294	4.3	1371007	1800.1	1684	633	12.5
G3-27002	9655	0.6	291	9.2	3346226	1800.2	3401	1211	12.7
G3-27003	9448	0.3	303	0.3	307	28.3	23	21	0.3
G3-27004	10162	0.5	295	6.7	3410843	1800.0	182	61	2.2
G3-27005	9793	0.5	296	12.5	60081	1800.0	5138	1282	21.0
G3-27006	9661	0.3	293	0.1	301	33.9	21	11	0.2
G3-27007	9535	0.2	298	0.2	696583	1800.1	10	8	1.4
G3-27008	9455	0.3	295	1.2	3283010	1800.1	183	92	2.0
G3-27009	9521	0.5	289	1.1	3324766	1800.1	332	98	4.0
Average	7944.4	0.3	245.4	2.7	1028682.0	896.1	885.5	369.6	4.8
Median	7911.5	0.3	243.5	0.9	7576.5	553.3	222.5	109.0	1.3
Max	10162.0	0.6	311.0	12.5	4456005.0	1800.2	5138.0	2296.0	26.5
Min	6079.0	0.1	189.0	0.0	193.0	9.4	0.0	4.0	0.0
GP1-01	-	-	-	85.9	-	-	-	-	-
GP1-02	-	-	-	1010.1	-	-	-	-	-
GP1-03	-	-	-	846.0	-	-	-	-	-
GP1-04	-	-	-	845.0	-	-	-	-	-
GP1-05	-	-	-	774.8	-	-	-	-	-
GP1-06	-	-	-	957.0	-	-	-	-	-
GP1-07	-	-	-	848.5	-	-	-	-	-
GP1-08	-	-	-	727.9	-	-	-	-	-
GP1-09	-	-	-	475.1	-	-	-	-	-
GP1-10	-	-	-	782.0	-	-	-	-	-
GP1-11	-	-	-	854.1	-	-	-	-	-
GP1-12	-	-	-	889.9	-	-	-	-	-
GP1-13	-	-	-	923.6	-	-	-	-	-
GP1-14	-	-	-	664.5	-	-	-	-	-
GP1-15	-	-	-	986.7	-	-	-	-	-
GP1-16	-	-	-	822.6	-	-	-	-	-
GP1-17	-	-	-	837.3	-	-	-	-	-
GP1-18	-	-	-	877.5	-	-	-	-	-
GP1-19	-	-	-	984.9	-	-	-	-	-
GP1-20	-	-	-	775.8	-	-	-	-	-
Average	-	-	-	798.5	-	-	-	-	-
Median	-	-	-	845.5	-	-	-	-	-
Max	-	-	-	1010.1	-	-	-	-	-
Min	-	-	-	85.9	-	-	-	-	-
GP2-21	19579	0.4	3	856.6	-	-	-	-	-
GP2-22	51145	0.0	4	810.7	-	-	-	-	-
GP2-23	27388	0.0	5	957.5	-	-	-	-	-
GP2-24	23087	0.0	6	915.6	-	-	-	-	-
GP2-25	51876	0.0	5	814.5	-	-	-	-	-
GP2-26	19473	0.0	4	827.6	-	-	-	-	-
GP2-27	20394	0.0	6	738.7	-	-	-	-	-
GP2-28	54286	0.0	4	802.5	-	-	-	-	-
GP2-29	30249	0.0	6	976.2	-	-	-	-	-
GP2-30	21169	0.0	4	727.2	-	-	-	-	-
GP2-31	35305	0.0	5	834.6	-	-	-	-	-
GP2-32	30744	0.0	4	818.9	-	-	-	-	-
GP2-33	26077	0.0	3	767.3	-	-	-	-	-
GP2-34	27487	0.0	3	849.2	-	-	-	-	-
GP2-35	22731	0.0	4	1118.9	-	-	-	-	-
GP2-36	18497	0.0	4	729.1	-	-	-	-	-
GP2-37	29485	0.0	3	1061.9	-	-	-	-	-
GP2-38	27833	0.0	6	775.5	-	-	-	-	-
GP2-39	21346	0.0	3	765.9	-	-	-	-	-
GP2-40	27264	0.0	3	875.3	-	-	-	-	-
Average	29270.8	0.0	4.3	851.2	-	-	-	-	-
Median	27326.0	0.0	4.0	823.3	-	-	-	-	-
Max	54286.0	0.4	6.0	1118.9	-	-	-	-	-
Min	18497.0	0.0	3.0	727.2	-	-	-	-	-
Global Average	4198.9	1.8	128.6	271.8	530276.9	953.0	841.9	352.5	141.5

Global Median	6079.0	0.6	189.0	1.5	9176.0	856.2	23.0	17.0	1.4
Global Max	10162.0	6.1	311.0	1010.1	4456005.0	1801.2	6508.0	2439.0	1800.0
Global Min	152.0	0.1	5.0	0.0	0.0	9.4	0.0	0.0	0.0

The CCDA heuristic reaches the optimal solution values for 29 out of 39 instances from the literature. The heuristic fails to find feasible solutions to the instances of Groups 1 and 3 and, for these of Group 2, CCDA does not reach any optimal solution value.

Regarding the DP algorithm, it fails to find the optimal solution for all instances of Group 3 in the imposed time limit. Our implementation employs the A^* extraction policy, which is recognized for its superior performance in the literature (Ferone et al., 2021). To obtain the values $\pi(u,t)$ for all $u \in V$ we run Dijkstra's algorithm in the reverse network, starting from t. The reverse network of D is a digraph $D' = (V, A')$ for $A' = \{(v,u) \mid \forall\ (u,v) \in A\}$. The arcs in the reverse network preserve the same costs as the original arcs in D. The stopping criterion for the DP algorithm, in addition to the time limit, is to reach the known optimal solution for the instance.

As pointed out by Ferone et al. (2021), Table 3 reveals that, on average, the DP algorithm runs in negligible time for random and grid digraphs. These benchmark instances have proven to be quite manageable for the DP algorithm. However, when it comes to the new instances, we observe that the average computational time is significantly higher than that w.r.t. the benchmark instances from the literature. Notably, the DP algorithm exhibits a 6.6% increase in cpu time for Group 2 in comparison to Group 1. Due to memory limitations and time constraints, the DP algorithm either runs out of memory or reaches the time limit for all instances within Group 3. Because of this, we do not present results for this group.

Concerning the B&B algorithm, we adopt the BFS node evaluation policy in the search tree, since it presents the best numerical results (Ferone et al., 2019). The B&B algorithm was not able to find an optimal solution in the time limit for any instances of Groups 1, 2, and 3. The algorithm failed to discover an optimal solution for 7 out of 19 random instances and for 11 out of 20 grid instances. We remark that the instance reduction algorithm helps to improve the number of proven optimal solutions compared to the results of Ferone et al. (2019), where the B&B algorithm found the optimal solution for 10 (resp. 8) random (resp.

grid) instances.

With respect to the B&C based formulation (PCM), if a path does not violate the limit k on the number of colors, then it is optimal to the k-CSPP. Otherwise, we add the corresponding violated cut (7) to the current model (PCM) and solve it with the B&C module of the IBM CPLEX solver. We use lazy callbacks to cut paths violating inequalities (7). The stopping criterion is based on reaching either the time limit or obtaining the optimal solution for the k-CSPP.

Excepting for three instances from the literature, the remaining ones were solved to optimality. Three random instances (R1-27199, R1-27204, and R1-27205) reached the time limit (they did not benefit significantly from the reduction algorithm) with a CPLEX lower bound far from their optimal solution values of 39.62%, 34.08%, and 40.97%, respectively. For the remaining 16 random instances, the model run with an average cpu time of 1.4 seconds and an average number of cuts (7) of 9.4. For grid digraphs, the B&C approach solved all of them to optimality. They presented average values of cpu time, cuts (7), and CPLEX B&C nodes, of 4.8 seconds, 369.6 cuts, and 885.5 nodes, respectively. We remark that 9 instances (8 random and 1 grid) were solved at the root node of the B&C tree with no generation of cuts (7). Concerning the new instances of Groups 1, 2, and 3, solving them with the B&C approach for model (PCM) was not possible as the Ubuntu operating system aborted all executions of the CPLEX solver for these groups of instances.

6.2. Results for model (FFP) with valid inequalities

In this section, we discuss the impact of the valid inequalities from Section 3 for model (FFP). In Table 4, we report computational results for random and grid instances (Ferone et al., 2019), and the new ones from Groups 1, 2, and 3. The first column *Ineq* indicates the set of inequalities we add to the model (FFP). In the initial set of rows, we display various statistics for each set of test-bed instances in the second column. This includes the average values of the optimal solutions denoted as *opt*, along with the average number of colors in these solutions. Additionally, we provide data on the average cost of the shortest paths, labeled as *sp*, for these instances, along with the average number of colors in these shortest paths, represented as *spc*.

For each combination of valid inequalities we add to model (FFP), from the second to the last group of rows, we present their impact on the average linear relaxed value z_r and the corresponding average cpu_r time; the average *cpu* time[1] to solve these instances and the average number of CPLEX B&B nodes *bb*; and finally, when inequalities (18) are present, we also present the average number of *cuts* added to model (FFP) by a CPLEX user-cut callback. Average execution times equal to zero means values less than 0.05 seconds. We separate results for distinct groups of inequalities by a horizontal line. For instance, we report results for the set of constraints defining model (FFP) in the second group of rows. The second set of experiments is for model (FFP) with the addition of inequalities (19). We show in bold the best results for z_r, *cpu*, and *bb* for the new instances. The reader is referred to a supplemental material for individual results for all test-bed instances.

With regard to the benchmark instances (Ferone et al., 2019) in Table 4, we observe that the valid inequalities cannot improve their average linear relaxed values and present small differences in both cpu_r and cpu times w.r.t. model (FFP). All these instances were solved at the root node of the B&B search tree ($bb = 0.0$), with their average shortest path cost being very close to the optimal linear relaxed and integer values (for random instances, $opt = 255.8$ and $sp = 201.8$, while for grid ones, $opt = 7936.9$ and $sp = 7929.1$). This also occurs for the average number of distinct colors in the shortest path solutions (for random instances, $colors = 5.7$ and $spc = 8.1$, while for grid ones, $colors = 246.3$ and $spc = 248.5$). On the other hand, for the new instances, on average, the shortest path *sp* is far from the optimal solution values *opt*. This is also true when comparing colors with *spc* of these instances.

In Table 4, we note that our instances require more average cpu times than the benchmark ones, despite their original dimensions. Furthermore, their average linear relaxation gaps ($100(opt - z_r)/z_r$), in percentage, are 119.8%, 112%, and 100% for Groups 1, 2, and 3, respectively. When examining the individual application of distinct sets of inequalities,

[1] For the instances of Group 3, model (FFP) fails to find the optimal solution for 2 instances, while the models (FFP)+(10)(11), (FFP)+(15), (FFP)+(16), (FFP)+(17), (FFP)+(19), and finally (FFP)+(15)(17)(18)(19) fail to solve exactly one instance each.

Table 4: The impact of the valid inequalities for model (FFP) for benchmark (Ferone et al., 2019) and the instances of Groups 1, 2, and 3.

Ineq		Random	Grid	Group 1	Group 2	Group 3
Average features	*opt*	255.8	7936.9	6083.1	6229.3	5977.2
	sp	201.8	7929.1	1007.0	1109.4	1080.3
	colors	5.7	246.3	7.0	6.7	7.7
	spc	8.1	248.5	15.7	16.1	17.0
(FFP)	z_r	253.5	7934.9	2767.9	2938.4	2988.4
	cpu_r	2.9	0.1	0.1	0.1	0.1
	cpu	2.9	0.1	600.0	491.3	864.2
	bb	0.0	0.0	29577.4	18095.1	42479.2
+(19)	z_r	253.5	7934.9	3098.9	3261.2	3115.1
	cpu_r	2.9	0.1	0.0	0.0	0.0
	cpu	2.9	0.1	110.6	110.4	413.9
	bb	0.0	0.0	4308.3	4363.6	12027.2
+(18)	z_r	253.5	7934.9	3219.3	3365.5	3251.0
	cpu_r	2.5	0.0	0.2	0.2	0.2
	cpu	2.9	0.2	115.1	182.1	435.4
	bb	0.0	0.0	1853.6	2804.8	8430.6
	cuts	0.0	0.0	20436.5	36680.0	35234.7
+(17)	z_r	253.5	7934.9	2991.3	3155.8	3023.0
	cpu_r	2.9	0.1	0.1	0.1	0.1
	cpu	2.9	0.1	246.8	200.2	677.6
	bb	0.0	0.0	10797.7	7766.3	30258.5
+(16)	z_r	253.5	7934.9	2955.6	3122.0	3018.8
	cpu_r	2.9	0.1	0.1	0.1	0.1
	cpu	2.9	0.1	295.1	222.9	742.2
	bb	0.0	0.0	12224.1	8686.2	33156.6
+(15)	z_r	253.5	7934.9	3106.9	3264.8	3057.7
	cpu_r	2.8	0.1	0.1	0.1	0.1
	cpu	2.8	0.1	113.8	114.6	634.7
	bb	0.0	0.0	4257.9	4657.5	28757.6
+(12)(13)(14)	z_r	253.5	7934.9	3098.6	3255.6	3052.0
	cpu_r	3.3	0.1	0.1	0.1	0.1
	cpu	3.3	0.1	154.4	132.9	580.2
	bb	0.0	0.0	5214.5	5221.5	27917.5
+(10)(11)	z_r	253.5	7934.9	2790.6	2958.1	3016.5
	cpu_r	3.0	0.1	0.1	0.1	0.1
	cpu	3.0	0.1	541.9	432.0	732.0
	bb	0.0	0.0	24581.0	15902.6	35159.2
+(15)(17)(18)	z_r	253.5	7934.9	**3297.6**	**3440.4**	3284.6
	cpu_r	2.5	0.0	0.2	0.2	0.1
	cpu	3.0	0.1	72.2	83.3	**303.7**
	bb	0.0	0.0	1641.5	1698.3	**5088.6**
	cuts	0.0	0.0	13707.0	15739.4	18663.2
+(17)(18)(19)	z_r	253.5	7934.9	3288.3	3435.6	3286.5
	cpu_r	0.0	0.0	0.0	0.0	0.0
	cpu	2.7	0.2	**45.8**	**76.4**	334.8
	bb	0.0	0.0	**1117.2**	**1630.5**	7185.3
	cuts	0.0	0.0	9288.3	14656.4	30521.2
+(15)(17)(18)(19)	z_r	253.5	7934.9	**3297.6**	**3440.4**	**3288.8**
	cpu_r	2.5	0.0	0.4	0.5	0.4
	cpu	2.9	0.2	87.4	116.6	467.1
	bb	0.0	0.0	1261.8	1682.6	5378.1
	cuts	0.0	0.0	17448.7	24721.3	48055.2

set (18) yielded the most favorable outcomes across these three groups in terms of linear relaxation and the number of B&B nodes. Conversely, the set of inequalities (19) delivered the most efficient cpu times. Indeed, for Groups 1, 2 and 3, we observe an improvement on cpu times in comparison to the one of model (FFP) of up to 81.57%, 77.52% and 52.10%, respectively. Concerning the combined use of valid inequalities, the models (FFP)+(15)(17)(18)

and (FFP)+(15)(17)(18)(19) both obtain, on average, the strongest linear relaxed values for Groups 1 and 2, while the latter obtained the best results for Group 3. For the number of B&B nodes, the best result is attained by model (FFP)+(17)(18)(19) for Groups 1 and 2, while model (FFP)+(15)(17)(18) reached the smallest number of B&B nodes and cpu times for Group 3. We highlight that the linear relaxation gaps for model (FFP)+(15)(17)(18)(19) are 84.5%, 81%, and 81.7% for Groups 1, 2, and 3, respectively.

We remark that all proposed inequalities for model (FFP) improve its linear relaxation when handling the new instances. Since inequalities (15) dominate the ones (12)–(14), which, in turn, dominate inequalities (10) and (11), as well as inequalities (17) dominate the ones (16), we do not report results for models combining these dominated inequalities. With respect to execution times for instances of Groups 1 and 2, compared to those obtained by the DP algorithm, model (FFP)+(17)(18)(19) provides an improvement of 94.26% and 91.02%, respectively. We emphasize that the DP algorithm is unable to find optimal solutions for instances of Group 3 within the time limit, while this model obtained the optimal solution for all these instances. For grid and random instances, the DP procedure presents an increase of 96.15% and an improvement of 96.29% of execution times, respectively, in comparison to this model.

Table 5: Statistics of the relative integrality gap (ratio) between optimal and linear relaxed solutions of variations of model (FPP) for Groups 1, 2, and 3.

	Ratios											
	Group 1				Group 2				Group 3			
Ineq	Mean	Median	Max	Min	Mean	Median	Max	Min	Mean	Median	Max	Min
(FFP)	119.8	118.3	190.5	51.1	112.6	106.5	172.1	66.7	98.9	89.0	173.0	62.1
+(19)	96.2	96.3	153.0	37.8	91.5	84.1	149.3	53.8	90.3	83.4	164.6	55.9
+(18)	88.6	90.0	137.1	36.7	85.5	79.5	135.3	46.3	82.8	78.3	147.8	50.6
+(17)	103.4	100.8	163.2	41.1	98.1	90.6	159.7	56.7	93.0	85.8	163.0	61.1
+(16)	105.9	103.7	165.6	43.3	100.2	91.8	163.3	58.5	93.4	87.3	161.7	60.8
+(15)	95.7	96.1	152.4	37.4	91.3	84.0	149.3	53.8	94.9	84.7	170.4	58.3
+(12)(13)(14)	96.2	96.5	153.0	37.9	91.8	84.3	150.4	54.3	95.3	86.2	170.4	55.8
+(10)(11)	118.1	116.3	189.8	50.9	111.3	104.9	172.1	66.0	97.6	89.7	172.7	61.9
+(15)(17)(18)	**84.1**	**83.9**	133.1	33.1	**81.6**	**76.1**	132.4	43.9	81.0	75.1	147.8	50.4
+(17)(18)(19)	84.6	84.3	133.5	33.1	81.8	**76.1**	132.4	43.9	80.8	75.1	147.8	50.4
+(15)(17)(18)(19)	**84.1**	**83.9**	133.1	33.1	**81.6**	**76.1**	132.4	43.9	**79.8**	**74.1**	147.8	50.4

Table 5 reports statistics results for Groups 1,2, and 3, as "Mean", "Median", maximum "Max", and minimum "Min" ratio values relative to the integrality gap ($100(opt - z_r)/z_r$) of the instances when solved with the use of valid inequalities for model (FPP). We highlight, in bold, the smallest mean and median values for each group. We observe, considering the models from (FFP)+(19) to (FFP)+(10)(11), that inequalities (18) provide the smallest mean and median ratios for all the three groups. Concerning the joint use of valid inequalities in the last three lines of this table, we note a slight difference between the mean and median ratios of the three groups. Globally, model (FFP)+(15)(17)(18)(19) obtained the smallest mean and median ratios.

7. Conclusions

In this work, we proposed valid inequalities for the k-Color Shortest Path Problem (k-CSPP) and showed that they strengthen the existing formulation (FFP) (Ferone et al., 2019) for this problem. One of the exponential-size set of valid inequalities was explored as a B&C algorithm for the k-CSPP, referred to as model (PCM). We also reproduced the instance reduction procedure of Cerrone and Russo (2023) and pointed that the Dijkstra-based heuristic CCDA can fail finding a feasible solution for the k-CSPP depending on the penalties adopted by this heuristic.

We observed that CPLEX finds no difficulty in solving the benchmark instances (Ferone et al., 2019) at the root node of its B&B search tree with model (FFP). This, because the reduction procedure drastically reduces the large dimensions of almost all instances (excepting for three of them). Their linear relaxed and optimal solution values are very close to the solution of their shortest paths. This motivated us to propose three groups of more difficult instances for the problem. The CCDA heuristic fails finding a feasible solution for Groups 1 and 3 of the new instances, and the quality of the solutions obtained for the instances of the Group 2 are far from the optimal ones. Moreover, the reduction procedure was not able to reduce the number of arcs and vertices of these new instances. The values of the linear relaxed and optimal solution values are not close to the solution of the shortest paths for these instances.

Concerning the numerical results, for the new instances, inequalities (18) individually obtain the best improvement on the linear relaxation of model (FFP) as well as on the reduction of the number of evaluated CPLEX B&B nodes. On the other hand, inequalities (19) obtain the smallest cpu times to solve these instances. Considering the combined use of the proposed inequalities, models (FFP)+(15)(17)(18) and (FFP)+(15)(17)(18)(19) obtained the best results for Groups 1 and 2, while the latter model attained slightly better results for Group 3. We emphasize that the B&B procedure (Ferone et al., 2019) fails to find the optimal solution for all the new instances, while the DP algorithm (Ferone et al., 2021) fails to find the optimal solutions for the instances of Group 3. Although DP algorithm finds optimal solutions for Groups 1 and 2, it requires more computational time compared to the mathematical models. We remark that despite the improvement on the linear relaxed solutions, they are still far from their optimal solutions. Thus, indicating that further research can be done in this direction.

As future research, we intend to handle problems like the MCPP with the proposed inequalities, and investigate whether they can be further strengthened. It seems that maximal cliques of non-reachable arcs (or colors) can be used to obtain facets of the polyhedron associated with model (FFP). Additionally, exploring more complex graph structures to address the k-CSPP remains an ongoing research challenge. While layered digraphs represent a promising step in this direction, they introduce certain unreachability between vertices, which, notably, allowed us to evaluate the importance of our proposed inequalities.

Acknowledgements

The authors are grateful to the 'Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES)' (Finance Code 001), to 'Cearense Foundation for Scientific and Technological Development Support (FUNCAP)' (Grant BMD-0008-01460.01.15/21), and to the Brazilian National Council for Scientific and Technological Development (CNPq grant 303988/2021-5). We thank the anonymous referees for their valuable comments and suggestions.

Declaration of generative AI and AI-assisted technologies in the writing process.

During the preparation of this work the author(s) used ChatGPT in order to improve language and readability, with caution. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

References

Cerrone, C. and Russo, D. D. (2023). An exact reduction technique for the k-colour shortest path problem. *Computers & Operations Research*, 149:106027.

Chang, R.-S. and Shing-Jiuan, L. (1997). The minimum labeling spanning trees. *Information Processing Letters*, 63(5):277–282.

Craveirinha, J., Pascoal, M., and Clímaco, J. (2023). An exact approach for finding bicriteria maximally srlg-disjoint/shortest path pairs in telecommunication networks. *INFOR: Information Systems and Operational Research*, 61(3):399–418.

Dehouche, N. (2020). The k-interchange-constrained diameter of a transit network: a connectedness indicator that accounts for travel convenience. *Transportation Letters*, 12(3):197–201.

Dorninger, D. (1994). Hamiltonian circuits determining the order of chromosomes. *Discrete Applied Mathematics*, 50(2):159–168.

Eiben, E. and Kanj, I. (2020). A colored path problem and its applications. *ACM Trans. Algorithms*, 16(4):1–48.

Ferone, D., Festa, P., Fugaro, S., and Pastore, T. (2021). A dynamic programming algorithm for solving the k-color shortest path problem. *Optimization Letters*, 15(6):1973–1992.

Ferone, D., Festa, P., and Pastore, T. (2019). The k-color shortest path problem. In Paolucci, M., Sciomachen, A., and Uberti, P., editors, *Advances in Optimization and Decision Science for Society, Services and Enterprises: ODS, Genoa, Italy, September 4-7, 2019*, pages 367–376. Springer International Publishing, Cham.

Kumar, N. (2019). Computing a minimum color path in edge-colored graphs. In Kotsireas, I., Pardalos, P., Parsopoulos, K. E., Souravlias, D., and Tsokas, A., editors, *Analysis of Experimental Algorithms*, pages 35–50, Cham. Springer International Publishing.

Motwani, R. and Raghavan, P. (1995). *Randomized Algorithms*. Cambridge University Press.

Pevzner, P. A. (1995). Dna physical mapping and alternating eulerian cycles in colored graphs. *Algorithmica*, 13(1):77–105.

Santos, R. F., Andrioni, A., Drummond, A. C., and Xavier, E. C. (2016). Multicolour paths in graphs: Np-hardness, algorithms, and applications on routing in wdm networks. *Journal of Combinatorial Optimization*, 33(2):742–778.

Shen, L., Yang, X., and Ramamurthy, B. (2005). Shared risk link group (srlg)-diverse path provisioning under hybrid service level agreements in wavelength-routed optical mesh networks. *IEEE/ACM Transactions on Networking*, 13(4):918–931.

Yuan, S., Varma, S., and Jue, J. (2005). Minimum-color path problems for reliability in mesh networks. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 4, pages 2658–2669.

Supplemental material

Table 6: Statistics for average results for the instances of Group 1.

Ineq.	linear relaxation				cpu				bb			
	Mean	*Median*	*Max*	*Min*	*Mean*	*Median*	*Max*	*Min*	*Mean*	*Median*	*Max*	*Min*
(FFP)	2767.9	2756.9	3181.8	2314.4	600.0	507.6	1719.6	9.3	29577.4	18407.5	83716.0	363.0
+(19)	3098.9	3119.8	3652.8	2590.5	110.6	124.9	391.6	12.3	4308.3	4350.0	17390.0	363.0
+(18)	3219.3	3200.6	3897.6	2642.9	115.1	80.4	635.3	6.1	1853.6	1174.5	11730.0	53.0
+(17)	2991.3	3019.8	3510.8	2449.6	223.2	185.3	786.4	3.8	10797.7	7442.0	37225.0	130.0
+(16)	2955.6	2989.0	3479.7	2444.7	295.1	271.5	887.3	5.2	12224.1	9921.5	36441.0	144.0
+(15)	3106.9	3123.8	3661.8	2596.5	113.8	80.0	434.5	13.6	4257.9	3030.5	18477.0	306.0
+(12),(13),(14)	3098.6	3115.1	3652.7	2593.2	154.4	165.6	619.3	10.9	5214.5	4940.5	19951.0	268.0
+(10),(11)	2790.6	2795.4	3188.7	2340.0	541.9	366.0	1738.5	24.1	24581.0	14189.0	70962.0	941.0
+(15),(17),(18)	**3297.6**	3293.3	3965.2	2693.0	**72.2**	57.6	411.7	3.6	1641.5	1231.0	10203.0	68.0
+(17),(18),(19)	3288.3	3286.2	3957.2	2686.9	45.8	43.0	137.5	2.7	**1117.2**	1067.0	2825.0	43.0
+(15),(17),(18),(19)	**3297.6**	3293.3	3965.2	2693.0	87.4	51.5	422.4	2.9	1261.8	900.5	5329.0	45.0

Table 7: Statistics for average results for the instances of Group 2.

Ineq.	linear relaxation				cpu				bb			
	Mean	*Median*	*Max*	*Min*	*Mean*	*Median*	*Max*	*Min*	*Mean*	*Median*	*Max*	*Min*
(FFP)	2938.4	2826.8	3450.0	2656.1	491.3	416.6	1614.7	126.3	18095.1	18069.5	44310.0	4469.0
+(19)	3261.2	3206.7	3802.7	2880.4	110.4	104.8	351.1	11.8	4363.6	3342.0	13303.0	660.0
+(18)	3365.5	3329.1	3823.2	3011.4	182.1	97.0	580.4	19.8	2804.8	1867.5	8797.0	345.0
+(17)	3155.8	3070.6	3703.4	2765.8	200.2	195.8	462.3	26.2	7766.3	7319.0	16921.0	1611.0
+(16)	3122.0	3043.3	3664.0	2764.9	222.9	206.3	559.8	17.9	8686.2	8224.0	20859.0	1043.0
+(15)	3264.8	3221.5	3803.1	2880.4	114.6	110.8	318.5	11.2	4657.5	4023.0	11072.0	598.0
+(12),(13),(14)	3255.6	3204.6	3800.1	2880.4	132.9	118.7	283.5	21.0	5221.5	4197.5	11790.0	1351.0
+(10),(11)	2958.1	2851.8	3470.9	2663.5	432.0	342.2	1038.5	90.5	15902.6	14342.5	34393.0	2725.0
+(15),(17),(18)	**3440.4**	3424.9	3934.8	3055.9	83.3	68.8	306.1	10.6	1698.3	1427.0	5089.0	190.0
+(17),(18),(19)	3435.6	3406.8	3934.8	3054.1	**76.4**	59.5	283.5	13.4	**1630.5**	1273.0	5353.0	418.0
+(15),(17),(18),(19)	**3440.4**	3424.9	3934.8	3055.9	116.6	88.1	399.9	26.3	1682.6	1386.5	5295.0	371.0

Table 8: Statistics for average results for the instances of Group 3.

Ineq.	linear relaxation				cpu				bb			
	Mean	*Median*	*Max*	*Min*	*Mean*	*Median*	*Max*	*Min*	*Mean*	*Median*	*Max*	*Min*
(FFP)	2988.4	3026.6	3611.3	2423.5	864.2	714.2	1800.0	191.3	42479.2	39387.0	143871.0	5414.0
+(19)	3115.1	3073.3	3720.1	2485.8	413.9	235.1	1800.0	29.8	12027.2	8342.5	32957.0	1114.0
+(18)	3251.0	3167.1	3850.2	2741.3	435.4	418.6	1193.0	31.9	8430.6	6721.0	24230.0	602.0
+(17)	3023.0	3034.1	3614.0	2462.8	677.6	434.3	1800.0	130.1	30258.5	15243.5	119345.0	4888.0
+(16)	3018.8	3035.5	3611.5	2451.6	742.2	501.3	1800.0	43.0	33156.6	19594.5	101368.0	1870.0
+(15)	3057.7	3059.1	3614.0	2466.4	634.7	509.1	1800.0	25.4	28757.6	17909.5	123236.0	1118.0
+(12),(13),(14)	3052.0	3058.9	3690.7	2456.3	580.2	430.1	1463.4	38.3	27917.5	15128.5	111814.0	2031.0
+(10),(11)	3016.5	3028.5	3614.0	2430.2	732.0	394.8	1800.0	49.5	35159.2	13177.0	115724.0	2215.0
+(15),(17),(18)	3284.6	3181.5	3857.8	2745.7	**303.7**	252.6	1061.4	33.8	**5088.6**	3950.0	15973.0	882.0
+(17),(18),(19)	3286.5	3178.5	3857.8	2745.7	334.8	317.1	774.5	19.8	7185.3	6152.0	19518.0	616.0
+(15),(17),(18),(19)	**3288.8**	3181.5	3857.8	2745.7	467.1	424.9	1800.0	27.9	5378.1	4986.0	11947.0	718.0

Table 9: The impact of the valid inequalities for model (FFP) with the random digraphs (Ferone et al., 2019).

inequalities	solution	R1-27190	R1-27191	R1-27195	R1-27197	R1-27199	R1-27200	R1-27202	R1-27203	R1-27204	R1-27205	R2-27001	R2-27004	R2-27005	R2-27007	R2-27008	R2-27010	R2-27012	R2-27015	R2-27018
		instances																		
(FFP)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.2	0.1	0.1	0.1	6.4	0.1	0.1	0.1	7.0	39.4	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
	cpu	0.2	0.1	0.1	0.1	6.4	0.1	0.1	0.1	7.0	39.4	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(19)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.1	0.1	0.1	0.1	6.7	0.1	0.1	0.1	8.1	39.6	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
	cpu	0.1	0.1	0.1	0.1	6.7	0.1	0.1	0.1	8.1	39.6	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(18)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.1	0.0	0.0	0.0	4.4	0.0	0.0	0.0	5.2	37.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	0.1	0.1	0.1	0.1	6.5	0.1	0.1	0.1	8.2	39.5	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(17)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.1	0.1	0.1	0.1	6.0	0.1	0.1	0.1	8.0	39.2	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
	cpu	0.1	0.1	0.1	0.1	6.0	0.1	0.1	0.1	8.0	39.2	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(16)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.1	0.1	0.1	0.1	6.4	0.1	0.1	0.1	8.1	39.3	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
	cpu	0.1	0.1	0.1	0.1	6.4	0.1	0.1	0.1	8.1	39.3	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(15)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.1	0.1	0.1	0.1	5.8	0.1	0.1	0.1	9.0	37.4	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
	cpu	0.1	0.1	0.1	0.1	5.8	0.1	0.1	0.1	9.0	37.4	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(12),(13),(14)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.1	0.1	0.1	0.1	6.4	0.1	0.1	0.1	8.1	47.0	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
	cpu	0.1	0.1	0.1	0.1	6.4	0.1	0.1	0.1	8.1	47.0	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(10),(11)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.1	0.1	0.1	0.1	5.9	0.1	0.1	0.1	7.9	40.6	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
	cpu	0.1	0.1	0.1	0.1	5.9	0.1	0.1	0.1	7.9	40.6	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(15),(17),(18)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.0	0.0	0.0	0.0	5.1	0.0	0.0	0.0	5.1	37.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	0.1	0.1	0.1	0.1	6.4	0.1	0.1	0.1	9.0	39.7	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(17),(18),(19)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	0.1	0.1	0.1	0.1	6.2	0.1	0.1	0.1	7.9	35.1	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
+(15),(17),(18),(19)	r	240.5	201.0	152.0	252.0	333.0	236.0	238.5	255.0	401.0	426.0	289.0	246.0	192.7	231.0	190.0	246.0	244.0	238.0	205.0
	cpu_r	0.0	0.0	0.0	0.0	5.2	0.0	0.0	0.0	5.2	37.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	0.1	0.1	0.1	0.1	5.8	0.1	0.1	0.1	9.0	39.4	0.1	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2

Table 10: The impact of the valid inequalities for model (FFP) with the grid digraphs (Ferone et al., 2019).

inequalities	solution	G1-27000	G1-27001	G1-27002	G1-27003	G1-27004	G1-27005	G1-27006	G1-27007	G1-27008	G1-27009	G3-27000	G3-27001	G3-27002	G3-27003	G3-27004	G3-27005	G3-27006	G3-27007	G3-27008	G3-27009
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
(FFP)	cpu_r	0.4	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.4
	cpu	0.4	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.4
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(19)	cpu_r	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	cpu	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(18)	cpu_r	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.2
	cpu	1.5	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(17)	cpu_r	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.4
	cpu	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.4
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(16)	cpu_r	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	cpu	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(15)	cpu_r	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	cpu	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(12),(13),(14)	cpu_r	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.4
	cpu	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.4
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(10),(11)	cpu_r	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	cpu	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(15),(17),(18)	cpu_r	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.2
	cpu	0.3	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(17),(18),(19)	cpu_r	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	1.5	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.4
	r	6120.0	6233.0	6327.0	6195.5	6375.0	6077.3	6106.3	6197.0	6193.0	6181.0	9805.7	9783.0	9651.7	9446.5	10147.4	9793.0	9653.3	9535.0	9453.5	9424.0
+(15),(17),(18),(19)	cpu_r	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.2
	cpu	1.4	0.0	0.1	0.3	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.2	0.0	0.2	0.0	0.0	0.5

Table 11: The impact of the valid inequalities for model (FFP) with the instances of Group 1.

		instances																			
inequalities	solution	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
(FFP)	z	3181.8	2701.4	3049.4	2761.4	3147.2	2749.8	3016.0	2399.3	2704.1	2879.0	2713.9	2752.5	2852.8	2788.1	2933.4	2525.7	2794.0	2401.3	2314.4	2692.5
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	108.9	502.6	615.4	239.5	512.6	191.9	1213.0	493.2	9.3	1566.5	410.7	959.7	264.8	224.6	629.8	160.8	1719.6	680.5	971.8	524.4
	bb	2599.0	19084.0	34839.0	7775.0	17292.0	7968.0	67975.0	17731.0	363.0	76247.0	17214.0	55712.0	7675.0	5795.0	39503.0	5343.0	83716.0	39699.0	57550.0	27468.0
+(19)	z	3652.8	2963.0	3434.6	3109.0	3466.4	3139.8	3273.0	2733.3	2966.1	3161.9	3131.4	3029.3	3081.0	3054.5	3232.5	3130.5	3188.1	2590.5	2597.4	3042.5
	cpuz	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	20.2	121.0	128.8	13.0	153.3	140.7	69.9	179.4	12.3	224.4	44.8	153.3	138.3	20.2	138.1	21.5	391.6	22.1	186.1	34.0
	bb	449.0	5013.0	4975.0	695.0	4402.0	5864.0	3083.0	6392.0	363.0	8642.0	2352.0	5372.0	4298.0	868.0	4945.0	1056.0	17390.0	1455.0	6648.0	1903.0
+(18)	z	3897.6	3043.1	3501.3	3173.4	3556.6	3215.1	3482.5	2864.4	2988.0	3385.3	3273.2	3162.8	3208.4	3192.8	3393.2	3049.1	3420.2	2642.9	2776.7	3158.6
	cpuz	0.2	0.1	0.1	0.2	0.2	0.1	0.1	0.2	0.1	0.2	0.2	0.2	0.2	0.2	0.1	0.1	0.2	0.1	0.2	0.1
	cpu	33.6	87.3	41.2	44.9	341.3	33.3	114.8	133.3	6.1	635.3	111.3	72.9	86.9	49.2	97.2	22.7	191.9	73.9	88.6	35.8
	bb	249.0	1667.0	990.0	900.0	3639.0	736.0	1970.0	1995.0	53.0	11730.0	2078.0	1178.0	1170.0	540.0	1565.0	469.0	3017.0	1171.0	1302.0	653.0
	cuts	4795.0	15594.0	10697.0	10160.0	48039.0	8202.0	22986.0	21202.0	1278.0	116210.0	27665.0	12567.0	14465.0	8162.0	14924.0	7235.0	28670.0	13239.0	14900.0	7739.0
+(17)	z	3510.8	2929.7	3218.9	3044.1	3365.9	3048.7	3184.6	2579.6	2894.9	3032.6	3079.3	2951.7	3007.0	2959.9	3072.9	2964.2	3071.1	2517.9	2449.6	2943.1
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	40.0	369.7	190.9	16.0	277.1	145.5	259.2	254.5	3.8	520.0	179.7	161.3	149.1	19.1	253.1	27.4	786.4	43.1	384.2	384.3
	bb	1079.0	22911.0	7655.0	1000.0	15311.0	6089.0	11656.0	10355.0	130.0	28082.0	7229.0	6649.0	5776.0	921.0	10092.0	1506.0	37225.0	2585.0	22564.0	17139.0
+(16)	z	3479.7	2891.2	3186.8	3023.2	3344.6	3002.2	3156.7	2549.1	2852.1	3009.2	3026.4	2932.1	2975.8	2933.7	3005.1	2868.2	3025.8	2489.7	2444.7	2915.5
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	50.4	233.4	183.8	162.9	309.5	366.7	484.7	477.0	5.2	588.5	340.3	46.3	205.6	28.4	316.3	26.2	887.3	141.6	606.8	441.7
	bb	1299.0	9081.0	7435.0	4199.0	13405.0	15133.0	21601.0	14270.0	144.0	28114.0	21641.0	2662.0	4542.0	1279.0	10762.0	1393.0	36441.0	5587.0	25850.0	19644.0
+(15)	z	3661.8	2963.0	3468.5	3116.2	3466.8	3139.8	3274.0	2733.8	2973.7	3169.3	3131.4	3038.3	3090.7	3071.0	3238.7	3153.6	3193.7	2596.5	2610.9	3045.8
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.1	0.1
	cpu	22.4	183.9	128.6	27.5	133.9	49.8	185.0	157.5	13.6	434.5	52.4	191.5	39.3	107.5	152.5	24.3	286.3	20.1	44.8	20.5
	bb	485.0	4812.0	4614.0	1055.0	4877.0	2112.0	6890.0	5654.0	306.0	18477.0	2564.0	6571.0	1695.0	3497.0	6119.0	1146.0	9623.0	1433.0	2112.0	1116.0
+(12),(13),(14)	z	3652.7	2960.8	3456.4	3108.1	3465.0	3139.8	3273.0	2733.3	2962.3	3168.0	3122.0	3038.3	3081.0	3054.5	3205.7	3122.1	3183.3	2593.2	2610.9	3042.2
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	21.3	170.6	127.9	26.3	162.1	131.5	233.9	190.3	10.9	231.6	188.2	197.3	35.2	29.3	169.1	17.6	619.3	30.1	176.2	319.4
	bb	511.0	4919.0	4728.0	953.0	4962.0	4212.0	6118.0	5805.0	268.0	11678.0	5927.0	7312.0	1927.0	1147.0	6021.0	762.0	19951.0	1404.0	5320.0	10364.0
+(10),(11)	z	3188.7	2706.1	3167.4	2801.1	3158.4	2757.2	3016.1	2422.0	2706.9	2901.4	2747.3	2804.8	2863.8	2789.7	2965.4	2558.4	2807.9	2416.4	2340.0	2692.5
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	104.7	533.0	521.7	46.5	291.0	361.9	1738.5	876.4	24.1	1341.9	370.2	335.6	164.0	267.0	499.0	228.1	1322.3	157.9	833.5	821.1
	bb	2299.0	19407.0	13979.0	2593.0	11079.0	19142.0	70285.0	54447.0	941.0	69503.0	13915.0	14399.0	6531.0	6988.0	18168.0	7273.0	70962.0	7594.0	45182.0	36932.0
+(15),(17),(18)	z	3965.2	3113.9	3594.3	3319.1	3600.5	3227.3	3537.3	2956.5	3069.7	3471.0	3333.9	3314.7	3250.0	3234.7	3461.4	3272.0	3454.6	2693.0	2861.6	3220.6
	cpuz	0.2	0.2	0.2	0.2	0.2	0.1	0.2	0.1	0.1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.1	0.2	0.2
	cpu	19.5	60.5	139.4	30.4	77.6	21.8	83.6	69.2	3.6	411.7	54.7	61.9	19.7	32.9	80.6	30.9	130.2	20.7	32.6	62.7
	bb	181.0	1538.0	2297.0	731.0	1781.0	618.0	1968.0	1708.0	68.0	10203.0	1251.0	1449.0	531.0	575.0	1856.0	643.0	2522.0	796.0	903.0	1211.0
	cuts	2838.0	10682.0	18062.0	7369.0	15867.0	5133.0	17615.0	13111.0	816.0	78225.0	11293.0	14313.0	4839.0	5860.0	15539.0	6808.0	20944.0	6620.0	7379.0	10827.0
+(17),(18),(19)	z	3957.2	3112.9	3555.2	3311.4	3599.5	3227.3	3537.0	2956.5	3069.5	3450.6	3330.9	3302.4	3239.9	3218.3	3440.6	3270.0	3447.3	2686.9	2834.2	3218.4
	cpuz	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	20.5	65.6	56.6	18.3	50.9	23.8	52.4	64.9	2.7	84.1	58.5	35.1	24.8	15.9	72.5	33.6	137.5	23.3	51.9	22.2
	bb	205.0	1616.0	1535.0	495.0	1212.0	748.0	1471.0	1602.0	43.0	2247.0	1312.0	922.0	621.0	329.0	1747.0	766.0	2825.0	762.0	1288.0	598.0
	cuts	3211.0	11531.0	11621.0	4434.0	10363.0	5852.0	11884.0	13668.0	607.0	14433.0	12940.0	7339.0	5668.0	3597.0	14044.0	9413.0	23592.0	6323.0	10380.0	4865.0
+(15),(17),(18),(19)	z	3965.2	3113.9	3594.3	3319.1	3600.5	3227.3	3537.3	2956.5	3069.7	3471.0	3333.9	3314.7	3250.0	3234.7	3461.4	3272.0	3454.6	2693.0	2861.6	3220.6
	cpuz	0.3	0.5	0.2	0.3	0.5	0.2	0.6	0.5	0.2	0.6	0.5	0.4	0.3	0.2	0.5	0.2	0.5	0.2	0.6	0.3
	cpu	26.2	112.5	40.9	23.1	106.0	31.9	86.5	115.9	2.9	422.4	143.2	78.7	42.8	23.8	147.7	24.0	179.2	36.9	53.7	49.3
	bb	185.0	1788.0	866.0	370.0	1550.0	636.0	1485.0	1773.0	45.0	5329.0	1692.0	1200.0	735.0	366.0	2009.0	385.0	2425.0	755.0	935.0	706.0
	cuts	4067.0	20324.0	11385.0	6436.0	21937.0	9624.0	19232.0	23681.0	920.0	65910.0	26121.0	17579.0	11420.0	6310.0	29970.0	6195.0	32756.0	11446.0	13375.0	10286.0

Table 12: The impact of the valid inequalities for model (FFP) with the instances of Group 2.

inequalities	solution	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
(FFP)	z	3130.8	3186.2	2981.9	2682.3	3310.5	3306.3	2684.3	3450.0	2656.1	2723.8	2789.5	2822.4	2831.1	3136.3	2801.5	3074.1	2665.6	2722.1	2729.8	3082.6
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	461.4	274.4	191.3	274.8	607.5	708.6	846.6	443.2	254.6	343.3	390.0	725.8	230.4	507.0	1614.7	126.3	672.1	280.6	361.8	511.3
	bb	21047.0	10130.0	7842.0	10978.0	21863.0	25157.0	30056.0	19156.0	12391.0	12640.0	17215.0	23159.0	9226.0	18924.0	44310.0	4469.0	27393.0	10629.0	15054.0	20263.0
+(19)	z	3374.4	3466.1	3277.3	3048.3	3522.6	3701.8	3042.6	3802.7	2978.3	3096.8	3085.7	3272.7	3140.6	3475.1	3058.2	3316.3	2880.4	3001.9	3059.8	3623.2
	cpuz	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	176.0	22.9	38.6	31.0	231.0	136.8	177.6	41.5	29.6	91.4	141.7	164.7	22.5	128.4	351.1	11.8	211.1	30.3	118.2	52.4
	bb	6214.0	1216.0	2113.0	1699.0	7423.0	4832.0	7552.0	2205.0	1760.0	2728.0	5484.0	6599.0	1332.0	4557.0	13303.0	660.0	9245.0	1665.0	3926.0	2758.0
+(18)	z	3652.5	3628.7	3389.5	3201.8	3696.1	3692.4	3041.6	3823.2	3121.0	3255.8	3146.1	3357.4	3300.8	3578.4	3239.8	3411.6	3035.6	3027.6	3011.4	3697.9
	cpuz	0.2	0.2	0.1	0.1	0.2	0.2	0.1	0.2	0.1	0.1	0.2	0.2	0.1	0.2	0.3	0.1	0.2	0.1	0.1	0.2
	cpu	94.0	74.8	51.4	64.9	448.4	137.8	148.7	100.0	29.5	24.1	115.9	539.9	67.4	388.2	580.4	77.5	458.3	81.3	19.8	139.9
	bb	1891.0	1164.0	1100.0	1349.0	5224.0	2442.0	2640.0	1844.0	702.0	345.0	2210.0	7791.0	1376.0	5563.0	8797.0	921.0	6493.0	1469.0	478.0	2297.0
	cuts	15747.0	16193.0	11245.0	16474.0	75547.0	26575.0	28691.0	18865.0	7086.0	6757.0	21411.0	115740.0	14689.0	87774.0	111130.0	17208.0	90530.0	17809.0	5991.0	28138.0
+(17)	z	3311.3	3417.6	3215.5	2974.7	3440.8	3539.8	2921.5	3703.4	2867.7	3039.3	2984.4	3097.7	3043.5	3364.2	2935.6	3217.8	2765.8	2840.4	3004.0	3430.9
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	240.1	160.4	144.5	206.1	236.8	298.3	302.8	185.5	29.7	110.2	263.3	287.5	26.2	172.5	462.3	125.7	294.1	130.5	100.3	227.5
	bb	8921.0	5655.0	6166.0	6478.0	10138.0	11052.0	11931.0	7766.0	1718.0	3082.0	9563.0	12406.0	1611.0	6872.0	16921.0	3881.0	13183.0	4753.0	4163.0	9066.0
+(16)	z	3254.5	3379.9	3186.6	2965.1	3428.5	3523.5	2917.0	3664.0	2843.6	3005.0	2938.5	3070.0	3016.7	3307.4	2896.1	3192.0	2764.9	2839.3	2881.3	3366.3
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	229.9	45.0	163.6	179.1	341.2	252.8	307.7	182.7	126.0	107.0	249.3	376.3	113.3	364.3	559.8	32.5	406.6	154.3	17.9	248.1
	bb	9127.0	2266.0	6287.0	5935.0	13005.0	10899.0	12692.0	7321.0	4549.0	3312.0	10347.0	16316.0	3824.0	12543.0	20859.0	1386.0	15807.0	5875.0	1043.0	10330.0
+(15)	z	3374.5	3468.9	3277.3	3048.3	3526.8	3704.6	3042.6	3803.1	2978.3	3096.8	3085.9	3280.1	3165.7	3475.1	3058.2	3316.3	2880.4	3001.9	3079.7	3632.5
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	163.6	20.7	107.5	27.7	318.5	50.0	197.3	58.9	24.4	14.4	177.5	215.8	15.8	42.2	259.7	11.2	206.4	114.1	120.7	144.4
	bb	6633.0	1079.0	4242.0	1449.0	9565.0	2641.0	7998.0	3044.0	1447.0	598.0	6445.0	9332.0	899.0	2379.0	11072.0	613.0	9104.0	3804.0	4866.0	5939.0
+(12),(13),(14)	z	3374.4	3458.7	3273.2	3048.3	3522.6	3693.5	3042.5	3800.1	2973.6	3086.7	3082.7	3268.5	3140.6	3475.0	3045.0	3316.0	2880.4	3001.9	3049.8	3578.3
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	231.9	29.3	31.5	109.9	190.4	55.4	214.0	152.4	21.0	113.2	159.7	241.9	90.4	124.1	283.5	110.6	193.7	27.4	112.4	165.4
	bb	8205.0	1470.0	1849.0	3848.0	6995.0	2963.0	8168.0	6059.0	1351.0	3495.0	6412.0	9636.0	2866.0	4547.0	11790.0	3500.0	8567.0	1581.0	3563.0	7565.0
+(10),(11)	z	3141.8	3186.5	2992.8	2738.0	3311.9	3327.0	2684.6	3470.9	2663.5	2737.4	2804.6	2846.9	2856.7	3181.1	2802.3	3087.1	2667.6	2741.3	2801.6	3117.7
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	739.7	274.3	177.0	355.9	531.1	464.3	710.6	328.4	163.6	90.5	378.5	989.2	161.2	305.3	1038.5	192.4	671.7	236.2	231.9	599.2
	bb	23188.0	10027.0	7971.0	14994.0	19612.0	19123.0	26272.0	13691.0	6400.0	2725.0	16746.0	29681.0	7314.0	13255.0	34393.0	5699.0	27376.0	9591.0	9347.0	20646.0
+(15),(17),(18)	z	3688.8	3715.1	3452.5	3260.8	3697.7	3793.5	3144.8	3934.8	3154.8	3310.6	3190.2	3473.1	3397.2	3653.9	3280.4	3491.5	3055.9	3082.5	3206.9	3823.5
	cpuz	0.2	0.2	0.1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
	cpu	107.7	18.5	36.7	74.0	105.9	87.4	86.9	267.7	28.3	19.6	63.6	99.9	16.8	53.0	306.1	10.6	130.7	25.9	32.9	95.0
	bb	2193.0	490.0	1017.0	1271.0	2116.0	1924.0	2095.0	5089.0	934.0	355.0	1583.0	2001.0	477.0	1055.0	5042.0	190.0	2754.0	789.0	723.0	1867.0
	cuts	16256.0	4406.0	8281.0	15491.0	19664.0	16241.0	17946.0	52088.0	6626.0	5258.0	11578.0	18369.0	3787.0	9971.0	47020.0	2937.0	25441.0	6245.0	7402.0	19780.0
+(17),(18),(19)	z	3688.6	3711.1	3449.6	3260.8	3696.1	3791.1	3144.6	3934.8	3154.8	3310.6	3190.2	3451.9	3364.0	3653.9	3280.4	3491.5	3054.1	3082.5	3202.8	3798.7
	cpuz	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	283.5	19.4	50.8	68.2	88.7	48.2	93.4	47.8	29.4	26.3	88.8	99.4	13.4	92.6	172.3	34.6	100.5	47.3	18.2	106.0
	bb	5353.0	500.0	1280.0	1222.0	1956.0	1266.0	2204.0	1261.0	817.0	418.0	2067.0	2428.0	423.0	1611.0	3382.0	594.0	2250.0	1098.0	463.0	2017.0
	cuts	44906.0	4483.0	11006.0	14852.0	17247.0	9689.0	17572.0	9880.0	6314.0	6459.0	15898.0	22870.0	3391.0	17021.0	26476.0	9708.0	19679.0	10079.0	4267.0	21331.0
+(15),(17),(18),(19)	z	3688.8	3715.1	3452.5	3260.8	3697.7	3793.5	3144.8	3934.8	3154.8	3310.6	3190.2	3473.1	3397.2	3653.9	3280.4	3491.5	3055.9	3082.5	3206.9	3823.5
	cpuz	0.6	0.3	0.3	0.5	0.6	0.5	0.6	0.5	0.2	0.2	0.6	0.6	0.2	0.5	0.6	0.3	0.6	0.4	0.5	0.6
	cpu	119.3	30.0	48.9	127.9	317.4	79.1	139.9	71.4	39.7	26.3	113.1	100.7	35.6	97.2	280.8	56.1	399.9	66.3	69.6	112.6
	bb	1763.0	543.0	956.0	1436.0	4756.0	1400.0	1987.0	1234.0	843.0	371.0	1842.0	1618.0	588.0	1373.0	3295.0	674.0	5295.0	1008.0	974.0	1695.0
	cuts	21932.0	6806.0	12105.0	27127.0	75211.0	16887.0	26657.0	14973.0	11510.0	9087.0	23044.0	22345.0	9212.0	22260.0	41546.0	15686.0	79845.0	14838.0	17705.0	25649.0

Table 13: The impact of the valid inequalities for model (FFP) with the instances of Group 3.

		instances																			
inequalities	solution	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
(FFP)	z	3112.0	3145.9	3526.6	2871.4	3611.3	3196.4	2870.1	3104.1	2423.5	3052.2	3392.0	2687.6	2616.2	2858.3	3056.3	2543.5	2581.1	2892.3	3226.2	3001.0
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	694.8	1126.5	1800.0	1558.5	727.1	1800.0	219.8	266.5	1004.7	412.5	249.3	349.4	1771.9	215.4	1073.0	701.3	191.3	927.4	1625.6	569.5
	bb	17709.0	50871.0	45082.0	60863.0	33727.0	120174.0	9518.0	10764.0	65201.0	13673.0	7260.0	12292.0	143871.0	5414.0	74474.0	45047.0	10802.0	47153.0	57894.0	17794.0
+(19)	z	3269.8	3331.0	3619.2	2962.1	3720.1	3300.0	3015.5	3256.3	2485.8	3167.2	3673.8	2843.5	2769.8	2970.7	3128.1	2706.5	2733.4	2988.2	3342.8	3018.5
	cpuz	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	275.1	382.4	1524.3	599.3	1800.0	845.0	244.2	187.5	29.8	203.3	66.3	152.6	250.3	29.8	226.1	208.3	169.3	186.7	534.1	363.9
	bb	7087.0	10172.0	29539.0	32957.0	24491.0	24612.0	8169.0	7169.0	1576.0	7358.0	2629.0	6542.0	11491.0	1114.0	8729.0	8516.0	6833.0	5127.0	23847.0	12585.0
+(18)	z	3497.0	3710.6	3676.4	3162.8	3850.2	3455.0	3052.2	3305.1	2741.3	3383.0	3738.7	2936.3	2936.8	3049.8	3171.4	2871.9	2749.9	3111.3	3490.9	3130.0
	cpuz	0.2	0.2	0.3	0.1	0.2	0.2	0.1	0.2	0.2	0.1	0.1	0.1	0.2	0.1	0.2	0.1	0.1	0.1	0.1	0.2
	cpu	885.6	456.9	1193.0	380.3	810.5	1047.8	80.8	496.9	31.9	87.3	92.7	274.2	457.4	34.3	768.8	305.2	77.1	43.5	526.5	656.9
	bb	14650.0	6858.0	17522.0	5741.0	13986.0	15214.0	1836.0	6584.0	602.0	2247.0	1549.0	4827.0	6903.0	604.0	23472.0	8089.0	2150.0	895.0	10653.0	24230.0
	cuts	87805.0	26981.0	123450.0	20609.0	72424.0	48648.0	4050.0	26563.0	1552.0	8269.0	6530.0	12619.0	22879.0	2941.0	109862.0	18213.0	4662.0	3480.0	26608.0	76548.0
+(17)	z	3130.7	3233.0	3575.5	2882.4	3614.0	3218.0	2949.8	3168.6	2462.8	3066.6	3479.9	2704.8	2649.4	2874.9	3084.2	2557.6	2649.4	2900.8	3255.2	3001.5
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	967.4	1292.9	1167.5	1800.0	1289.3	1800.0	142.9	525.0	130.1	242.3	197.8	290.3	813.4	237.2	496.9	340.8	298.7	371.6	792.4	354.6
	bb	39926.0	52432.0	41920.0	49142.0	48790.0	119345.0	5046.0	29150.0	4888.0	10423.0	6715.0	12301.0	29579.0	7475.0	14612.0	15034.0	12118.0	11554.0	79267.0	15453.0
+(16)	z	3157.3	3249.3	3575.5	2894.4	3611.5	3200.8	2870.1	3180.7	2451.6	3070.1	3497.8	2688.0	2635.4	2879.9	3084.2	2554.5	2600.0	2900.0	3274.4	3001.0
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	1488.1	1254.2	1113.4	1800.0	1099.6	1321.3	213.1	253.6	122.4	269.7	279.6	252.4	789.0	43.0	539.9	462.8	214.7	404.1	1241.1	1681.4
	bb	55122.0	55209.0	43012.0	43204.0	47607.0	96525.0	4775.0	8573.0	4427.0	8790.0	7420.0	7997.0	69426.0	1870.0	19547.0	19642.0	7954.0	14207.0	46457.0	101368.0
+(15)	z	3157.3	3254.5	3599.1	2898.7	3614.0	3234.9	3009.9	3235.3	2466.4	3080.9	3612.9	2758.5	2668.2	2926.1	3086.9	2566.5	2659.7	2967.8	3319.6	3037.3
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	776.1	1044.6	1203.1	780.4	1192.7	1338.8	283.6	262.2	43.2	157.3	236.4	395.5	481.0	25.4	658.9	290.3	189.7	537.2	998.5	1800.0
	bb	39711.0	44927.0	36094.0	38159.0	47240.0	45334.0	6235.0	7787.0	2513.0	6236.0	7294.0	17611.0	16099.0	1118.0	19966.0	13457.0	5551.0	18208.0	78375.0	123236.0
+(12),(13),(14)	z	3157.4	3257.5	3610.6	2898.9	3690.7	3223.9	2880.3	3187.1	2456.3	3080.9	3609.4	2747.7	2656.0	2973.8	3095.1	2569.7	2613.3	2971.8	3322.4	3037.0
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	1444.4	706.8	684.4	461.0	1105.7	1266.3	38.3	223.3	53.8	255.2	236.4	208.9	1463.4	262.8	597.3	484.0	310.9	332.9	1068.9	399.1
	bb	46692.0	16823.0	25771.0	13206.0	49178.0	85383.0	2031.0	8509.0	2440.0	14856.0	5639.0	5339.0	111814.0	7593.0	40443.0	16941.0	11625.0	6998.0	71667.0	15401.0
+(10),(11)	z	3157.8	3145.9	3547.9	2874.0	3614.0	3225.0	2990.5	3139.9	2430.2	3052.4	3439.5	2774.1	2705.1	2861.9	3056.3	2550.2	2635.9	2892.3	3232.9	3004.6
	cpuz	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	684.1	1772.8	321.8	1661.2	726.1	1800.0	49.5	312.7	245.7	243.1	364.4	181.5	1439.3	208.6	1280.7	409.3	196.9	380.4	1227.3	1134.3
	bb	25146.0	46772.0	9673.0	52232.0	31615.0	115724.0	2215.0	10804.0	7894.0	8556.0	8645.0	5013.0	98310.0	5618.0	81946.0	15550.0	8204.0	8554.0	84833.0	75879.0
+(15),(17),(18)	z	3548.3	3723.5	3719.1	3163.0	3857.8	3484.8	3191.6	3356.7	2745.7	3383.6	3824.5	3031.1	2954.3	3080.3	3171.4	2874.2	2809.9	3115.6	3521.6	3135.5
	cpuz	0.2	0.2	0.1	0.2	0.2	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	cpu	392.6	452.1	234.6	505.1	1061.4	653.8	187.6	219.9	33.8	221.2	61.4	63.4	366.7	52.7	437.2	114.3	77.0	270.7	330.6	338.7
	bb	5340.0	5302.0	3897.0	6362.0	15973.0	10635.0	2814.0	3960.0	882.0	3841.0	1223.0	1701.0	6217.0	1154.0	7435.0	3120.0	2192.0	3940.0	7838.0	7945.0
	cuts	25714.0	23990.0	17796.0	20901.0	101906.0	32878.0	14287.0	8260.0	2593.0	10723.0	4391.0	3926.0	23978.0	3954.0	22173.0	5866.0	5342.0	18986.0	14874.0	10726.0
+(17),(18),(19)	z	3535.0	3725.6	3708.5	3163.0	3857.8	3484.8	3185.6	3356.7	2745.7	3383.6	3824.5	3031.1	2987.6	3080.3	3171.4	2890.7	2826.0	3115.6	3521.5	3135.5
	cpuz	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	cpu	582.2	284.0	585.1	379.8	313.9	774.5	516.3	335.7	28.8	59.6	87.2	60.0	320.9	47.8	693.6	274.9	246.2	19.8	765.8	320.4
	bb	10694.0	3796.0	11475.0	6245.0	4952.0	19518.0	13972.0	5383.0	745.0	1486.0	1805.0	1380.0	7275.0	1179.0	17338.0	7020.0	6059.0	616.0	14882.0	7885.0
	cuts	64657.0	12225.0	65384.0	23061.0	20357.0	124120.0	69252.0	22002.0	2011.0	4683.0	6953.0	2864.0	14613.0	2584.0	94834.0	11817.0	19290.0	1555.0	37790.0	10373.0
+(15),(17),(18),(19)	z	3548.3	3725.6	3719.1	3163.0	3857.8	3484.8	3191.6	3356.7	2745.7	3383.6	3824.5	3031.1	3003.9	3080.3	3171.4	2890.7	2826.0	3115.6	3521.6	3135.5
	cpuz	0.9	0.4	1.0	0.7	0.3	0.6	0.1	0.2	0.2	0.3	0.2	0.2	0.5	0.4	0.5	0.2	0.2	0.2	0.6	0.3
	cpu	881.7	574.1	1800.0	679.7	476.5	953.4	27.9	116.5	83.9	88.6	157.5	104.6	797.3	314.7	531.7	395.8	101.9	59.5	743.5	454.1
	bb	11860.0	4848.0	11947.0	6413.0	5124.0	10950.0	718.0	2226.0	1227.0	1415.0	1487.0	1671.0	9407.0	2759.0	9757.0	6312.0	1786.0	788.0	10174.0	6692.0
	cuts	150203.0	35951.0	217393.0	51568.0	40621.0	74514.0	3677.0	9860.0	11907.0	12370.0	15371.0	12477.0	54425.0	49573.0	61567.0	35811.0	11435.0	8476.0	70230.0	33674.0